Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Міністерство освіти і науки України

Кваліфікаційна наукова праця
на правах рукопису

ШИМКОВИЧ ВОЛОДИМИР МИКОЛАЙОВИЧ

УДК 004.896

ДИСЕРТАЦІЯ

МЕТОДИ ТА ЗАСОБИ ПРОЕКТУВАННЯ АПАРАТНИХ КОМПОНЕНТІВ НЕЙРОМЕРЕЖЕВИХ СИСТЕМ КЕРУВАННЯ

Спеціальність 05.13.05 – комп'ютерні системи та компоненти

Подається на здобуття наукового ступеня кандидата технічних наук

Дисертація містить результати власних досліджень.
Використання ідей, результатів і текстів інших авторів
мають посилання на відповідне джерело

_____ В. М. Шимкович

Науковий керівник Кравець Петро Іванович,
кандидат технічних наук,
старший науковий співробітник

Київ – 2021

АНОТАЦІЯ

Шимкович В. М. Методи та засоби проектування апаратних компонентів нейромережевих систем керування. На правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук за спеціальністю 05.13.05 – комп'ютерні системи та компоненти. Робота виконана на кафедрі автоматики та управління в технічних системах Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського» Міністерства освіти і науки України. Захист відбудеться у спеціалізованій вченій раді Д 26.002.02 при Національному технічному університеті України «Київський політехнічний інститут ім. Ігоря Сікорського», Київ, 2021.

Дисертацію присвячено вирішенню задачі підвищення ефективності роботи компонентів нейромережевих систем керування, які дозволяють синтезувати нейромережеві системи керування, що функціонують та адаптуються в режимі реального часу з врахуванням специфіки завдань керування.

ШНМ можуть бути використані в таких об’єктах як робототехніка, керування безпілотними літальними апаратами, керуванні транспортними засобами, розпізнаванні образів, аналізі та прийнятті рішень в системах Інтернету Речей, керуванні космічними кораблями, військовою технікою та багатьма іншими різними сферами застосування в сучасних технологіях. У цих системах нейронні мережі можуть використовуватися для ідентифікації об'єктів, прогнозування стану об’єктів, розпізнавання, кластеризації, класифікації, аналізу великої кількості даних, що надходять з великою швидкістю від великої кількості пристроїв та датчиків тощо. ШНМ можуть бути використані для побудови регулюючих та коректуючих пристроїв, еталонних, адаптивних, номінальних та інверсно-динамічних моделей об'єктів, на основі яких здійснюється дослідження об’єктів, аналіз впливу збурень, що діють на об’єкт, визначення оптимального закону керування, пошук або обчислення оптимальної програми зміни керуючого впливу при зміні значень параметрів об’єкту та характеристик вхідних даних.

Динамічне розширення вбудованих систем, що потребують використання апаратних засобів зі ШНМ, для підвищення ефективності своєї роботи, визначають підвищені вимоги до цих апаратних засобів, їх швидкодії, точності та використанні обчислювального ресурсу.

Таким чином науково-технічна задача підвищення ефективності апаратної реалізації нейромережевих компонентів систем керування динамічними об'єктами, що забезпечують адаптацію та самоналагодження систем керування в реальному часі є актуальна.

Об'єктом дослідження є нейромережеві системи керування динамічними об'єктами.

Предметом дослідження є методи та технології автоматизованого проєктування інтелектуалізованих апаратних засобів нейромережевих систем керування динамічними об'єктами.

Метою дисертаційної роботи є підвищення ефективності нейромережевих систем керування за рахунок створення швидкодіючих компонентів, які дозволяють реалізовувати функції ідентифікації, адаптації та керування динамічними об'єктами в реальному часі.

Результати дослідження викладено у чотирьох розділах дисертації.

У першому розділі виконано аналіз галузей застосування апаратних засобів з ШНМ, основних структур нейромережевих систем керування динамічними об'єктами. На основі аналізу встановлено, що нейромережеві системи керування складаються з прямих та інверсних моделей об’єкта керування та компоненту їх адаптації.

На основі аналізу ШНМ встановлено, що на сьогоднішній день розроблено та досліджено декілька десятків типів ШНМ, але основними, принципово різними типами є три типи мереж, це RBF-мережі, динамічні мережі Хопфілда та мережі прямого розповсюдження. Данні ШНМ, завдяки своїм властивостям, можуть бути використані для подальшого проєктування на їх основі прямих та інверсних моделей об'єкта керування.

Проаналізовано існуючі на сьогодні методи та алгоритми навчання ШНМ. На основі аналізу встановлено, що використання генетичного алгоритму для навчання нейромережевих компонентів СК є найбільш оптимальним при апаратній реалізації.

Проведено аналіз сучасного стану програмних, апаратних і апаратно-програмних засобів реалізації ШНМ. В результаті проведеного аналізу встановлено, що засоби реалізації нейромережевих систем керування повинні орієнтуватися на широке застосування в промислових умовах, бути універсальними і гнучкими, функціонувати та навчатися в реальному часі, бути простими і дешевими, тому найбільш перспективними засобами можна вважати FPGA. Проведено огляд основних робіт по реалізації ШНМ засобами FPGA.

У другому розділі запропоновано метод проєктування нелінійних функцій активації штучного нейрону на FPGA. На базі запропонованого методу розроблено алгоритми апаратної реалізації штучного нейрона з сигмоїдальною фукцією активації та нейрона прихованого шару RBF-мережі з функцією активації Гауса. Проведено дослідження реалізованих штучних нейронів та ШНМ. Показано, що за рахунок розробленого методу та алгоритмів забезпечується значна оптимізація використаного ресурсу, збільшується швидкість обчислень апаратних блоків з ШНМ та їх точність в порівнянні з аналогами.

У третьому розділі розроблено технологію для побудови, дослідження та оцінки нейромережевих моделей багатовимірних об'єктів керування для їх подальшої реалізації. Розроблено метод проєктування апаратних компонентів, таких як пряма та інверсна модель об'єкта керування, нейромережевих систем керування, які є базовими для структурного синтезу систем керування, для обчислення векторів стану об'єкта і формування функції керування ним. Розроблено метод оптимізації вагових коефіцієнтів ШНМ за допомогою генетичного алгоритму при реалізації на FPGA, що дозволяє значно підвищити швидкість адаптації прямої та інверсної моделі об'єкта керування.

У четвертому розділі реалізовано та досліджено на основі розроблених компонентів: системи керування без зворотного зв'язку; системи керування з

зворотнім зв'язком; адаптивну систему керування із прямою та інверсною моделями об'єкта керування; адаптивну нейромережеву систему керування з еталонною моделлю; розроблено макет діючого нейроконтролера системи стабілізації рухомого об'єкта на обмеженій площині.

У додатках наведено інформацію про впровадження результатів дослідження (Додаток А) та інформацію про наукові публікації і апробацію отриманих результатів (Додаток Б).

Ключові слова: нейронна мережа, системи реального часу, генетичний алгоритм, системи керування, програмована логічна інтегральна схема.

ABSTRACT

Shymkovych V.M. Methods and means of designing hardware components of neural network control systems. On the rights of the manuscript.

The dissertation on competition of a scientific degree of the candidate of technical sciences on a specialty 05.13.05 – Computer systems and components. The work was performed at the Department of Automation and Control in Technical Systems of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute". The dissertation will be held at the specialized scientific council D 26.002.02 of the National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv, 2021.

The dissertation is devoted to solving the problem of increasing the efficiency of the components of neural network control systems, which allow to synthesize neural network control systems that function and adapt in real time taking into account the specifics of control tasks.

ANN can be used in facilities such as robotics, unmanned aerial vehicle control, vehicle control, pattern recognition, analysis and decision making in the Internet of Things, spacecraft control, military equipment and many other applications of modern technologies. In these systems, neural networks can be used to identify objects, predict the state of objects, recognize, cluster, classify, analyze large amounts of data coming at high speed from a large number of devices and

sensors, and more. ANN can be used to build control and correction devices, reference, adaptive, nominal and inverse-dynamic models of objects, based on which the study of objects, analysis of the impact of perturbations acting on the object, determining the optimal control law, search or calculation of the optimal program to change the control effect when changing the values of the parameters of the object and the characteristics of the input data.

Dynamic expansion of embedded systems that require the use of hardware with ANN, to increase the efficiency of their work, determine the increased requirements for these hardware, their speed, accuracy and use of computing resources.

Thus, the scientific and technical task of increasing the efficiency of hardware implementation of neural network components of control systems for dynamic objects, providing adaptation and self-tuning of control systems in real time is relevant.

The object of the research is neural network control systems for dynamic objects.

The subject of research is methods and technologies of automated design of intellectualized hardware of neural network control systems of dynamic objects.

The purpose of the dissertation is to increase the efficiency of neural network control systems by creating high-speed components that allow you to implement the functions of identification, adaptation and control of dynamic objects in real time.

The results of the research are presented in four sections of the dissertation.

Section 1 analyzes the areas of application of ANN hardware, the main structures of neural network control systems for dynamic objects. Based on the analysis, it was found that neural network control systems consist of direct and inverse models of the control object and the component of their adaptation.

Based on the ANN analysis, it has been established that several dozen types of ANN have been developed and researched to date, but the main, fundamentally different types are three types of networks: RBF-networks, Hopfield dynamic

networks and direct distribution networks. Due to the properties ANN, they can be used for further design on their basis of direct and inverse models of the control object.

The existing methods and algorithms of ANN learning are analyzed. Based on the analysis, it is established that the use of a genetic algorithm for training the neural network components of the CS is the most optimal for hardware implementation.

The analysis of the current state of software, hardware and hardware-software means of realization of ANN is carried out. As a result of the analysis it was established that the means of implementation of neural network control systems should be focused on widespread use in industrial conditions, be universal and flexible, function and learn in real time, be simple and cheap, so the most promising tools can be considered FPGA. The review of the main works on realization of ANN by means of FPGA is carried out.

Section 2 proposes a method for designing nonlinear activation functions of an artificial neuron on an FPGA. Based on the proposed method, algorithms for hardware implementation of an artificial neuron with sigmoidal activation function and a hidden layer neuron of the RBF network with a Gaussian activation function have been developed. A research of implemented artificial neurons and ANN was performed. It is shown that due to the developed method and algorithms significant optimization of the used resource is provided, the speed of calculations of hardware units with ANN and their accuracy in comparison with analogues increases.

Section 3 develops technology for construction, research and evaluation of neural network models of multidimensional control objects for their further implementation. A method of designing hardware components, such as direct and inverse model of control object, neural network control systems, which are basic for structural synthesis of control systems, for calculation of state vectors of object and formation of control function is developed. The method of optimization of ANN weighting factors by means of genetic algorithm at realization on FPGA is

developed that allows to increase considerably speed of adaptation of direct and inverse model of control object.

In the fourth section implemented and researched on the basis of the developed components: control systems without feedback; feedback control systems; adaptive control system with direct and inverse models of the control object; adaptive neural network control system with a reference model; a model of the operating neurocontroller of the stabilization system of a moving object on a limited plane is developed.

The appendices provide information on the implementation of the research results (Appendix «А») and information on scientific publications and approbation of the obtained results (Appendix «Б»).

Keywords: neural network, real-time systems, genetic algorithm, control systems, field-programmable gate array.

СПИСОК ПУБЛІКАЦІЙ ЗА ТЕМОЮ ДИСЕРТАЦІЙНОЇ РОБОТИ / THE LIST OF AUTHOR'S PUBLICATIONS FOR DISSERTATION THESES

Безпосередньо за тематикою дисертаційної роботи опубліковано 38 наукових робіт, а саме: /

Directly for the subject of dissertation theses 38 scientific publications have been published:

- статті в наукових фахових журналах України (10) /

- articles in Ukrainian professional journals (10):

1. Шимкович В. М., Дорошенко А. Ю., Федоренко В. О. Програмні засоби моделювання системи керування векторною тягою реактивного двигуна // Проблеми програмування. 2018. № 2-3. С. 296-304.

2. Шимкович В. Н., Кравец П. И., Ференс Д. А. Метод и алгоритмы реализации на ПЛИС функции активации для искусственных нейронных сетей // Международный научно-технический журнал «Электронное моделирование». 2015. Том 37, №4. С. 63-73.

3. Шимкович В. М., Кравець П. І., Федорчук В. В., Гой А. А. Нейромережевий контролер системи стабілізації рухомого об'єкта з апаратно-програмною реалізацією на ПЛІС // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2014. №63. С. 4-11.

4. Шимкович В. Н., Кравец П. И. Метод оптимизации весовых коэффициентов нейронных сетей с помощью генетического алгоритма при реализации на программируемых логических интегральных схемах // Международный научно-технический журнал «Электронное моделирование». 2013. Том 35, №3. С. 65-75.

5. Шимкович В. М., Кравець П. І., Омельченко П. В. Нейромережеві компоненти систем керування динамічними об'єктами з їх апаратно-

програмною реалізацією на FPGA // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2013. № 59. С. 78-85.

6. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Ткач І. І. Розробка та дослідження технології оцінювання показників нейромережевих моделей MIMO-об'єктів керування // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2012. №57. С. 144–149.

7. Шимкович В. М., Кравець П. І., Зубенко Г. А.Технологія апаратно-програмної реалізації штучного нейрона та штучних нейронних мереж засобами FPGA // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2012. №55. С. 174-180.

8. Шимкович В. Н., Кравец П. И., Лукина Т. Й., Жеребко В. А. Методика аппаратно-программной реализации адаптивного нейросетевого ПИД-регулятора на FPGA-кристалле // Международный научно-технический журнал «Проблемы управления и информатики». 2011. № 2. С. 130-136.

9. Шимкович В. М., Кравець П. І., Жеребко В. А., Шимкович В. М., Дьомін Р. Ю., Мостович А.В. Нейромережеві технології оперативного діагностування технічного стану рухомого складу // Збірник наукових праць Українського державного університету залізничного транспорту. 2011. №123. С. 119–123.

10. Шимкович В. М., Кравець П. І., Жеребко В. А. Методика апаратно-програмної реалізації однонейронного нейромережевого ПІД-регулятора на FPGA // Журнал «Вісник ВПІ». 2011. №3. С. 148-152.

- статті в закордонних фахових виданнях, які реферуються базою Scopus (3), в тому числі:

- articles in foreign professional journals referenced by the Scopus database (3), including:

- третього квартиля (Q3) (2) / the third quartile (Q3) (2):

11. Volodymyr Symkovych, Peter Kravets. Hardware Implementation Neural Network Controller on FPGA for Stability Ball on the Platform // Hu Z.,

Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education II. ICCSEEA 2019. Advances in Intelligent Systems and Computing. 2020. Vol. 938. Springer, Cham, Switzerland. pp. 247-256.

12. Volodymyr Symkovych, Artem Volokyta, Ivan Volokyta, Vladyslav Vasyliev. Research and Development of a Stereo Encoder of a FM-Transmitter Based on FPGA // Hu Z., Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education. ICCSEEA 2018. Advances in Intelligent Systems and Computing. Vol. 754. Springer, Cham, Switzerland. pp. 92-101.

- четвертого квартиля (Q4) (1) / the fourth quartile (Q4) (1):

13. Vladimir N. Shimkovich, Petr I. Kravets, Tatyana I. Lukina, Valeriy A. Zherebko. Methods of Hardware and Software Realization of Adaptive Neural Network PID Controller on FPGA-Chip // Journal of Automation and Information Sciences. 2011. Issue 4, Volume 43. Begell House, New York, USA. pp. 70-77.

За результатами досліджень також опубліковано статті в інших наукових журналах України та за кордоном (2) /

According to the results of investigations, the articles in other scientific Ukrainian and foreign journals are also published (2):

14. Volodymyr Shymkovych, Volodymyr Samotyy, Sergii Telenyk, Petro Kravets, Taras Posvistak. A real time control system for balancing a ball on a platform with FPGA parallel implementation // Technical Transactions. Poland. 2018. Vol. 5. 109-117.

15. Шимкович В. М., Кравець П. І., Николин О. І. Нейромережева система машинного бачення з апаратно-програмною реалізацією на ПЛІС // Науковий журнал «Молодий вчений». 2015. № 5(20). с. 47-50.

Результати, викладені в дисертації, було апробовано на 21 науковій конференції, зокрема опубліковано 23 матеріалів конференцій /

The results shown in the theses were validated at 21 scientific conferences, in particular, were published in 23 materials of the conferences:

- що реферується базою Scopus (3) /

- which are referenced by the Scopus database (3):

16. Volodymyr Shymkovych, Veronika Niechkina. The criterion for determining the buffering time of the measuring channel for smoothing the variable changes of the sensor signal // *2020 IEEE 7th International Conference on Energy Smart Systems (ESS)*, Kyiv, Ukraine. 2020. pp. 343-346.

17. Volodymyr Symkovych, Anatoliy Doroshenko, Vladyslav Fedorenko. Software means of modeling of the vector type of reactive engine control system // CEUR Workshop Proceedings. 2018. №2139. pp. 296-305.

18. Volodymyr Symkovych, Peter Kravets, Volodymyr Samotyy. Method and technology of synthesis of neural network models of object control with their hardware implementation on FPGA // Proceedings of 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. Bucharest, Romania. 2017. pp. 947-951.

- інші конференції (20) / other conferences (20):

19. Volodymyr Symkovych, Sergii Telenyk, Petro Kravets, Taras Posvistak. FPGA Implementation of the PID Algorithm for Real Time Ball Balancing on the Platform // Proceedings of The Fourth International Conference on “Automatic Control and Information Technology” (December 14-16, 2017). Krakow, Poland, 2017. pp. 160-169.

20. Volodymyr Symkovych, Petro Kravets, Zahar Yurchenko. Algorithm for the Implementation of Radial-Basic Neural Networks and their Activation Functions on the FPGA // Proceedings of The Fourth International Conference on “Au-

tomatic Control and Information Technology" (December 14-16, 2017). Krakow, Poland, 2017. pp. 122-129. (Іноземне видання)

21. Volodymyr Shymkovych, Petro Kravets. Neural Network Control System with Direct and Inverse Model of Control Object Hardware and Software Realization of in FPGA // Information Technology, Computational and Experimental Physics. Kulczycki P., Kowalski P.A., Lukasik S. (eds.) AGN-UST. Krakow, Poland. 2016. pp. 180-183.

22. Шимкович В.М. Кравець П.І., Николин О.І. Неромережева система комп'ютерного бачення // Наука України: проблеми сьогодення та перспективи розвитку: матеріали наук.-практ. конф. з міжнар. участю (м. Одеса, 29-30 травня 2015р.). Одеса, 2015. с. 225-227.

23. Шимкович В. М., Кравець П. І., Омельченко П. В. Програмне середовище і технологія моделювання нейромережевих систем керування динамічними об'єктами // Infocom Advanced Solutions 2015: матеріали І-ї міжнародної конференції присвяченої 70-річчю кафедри автоматики та керування в технічних системах (м. Київ, 24-25 листопада 2015 р.). Київ, 2015. с. 80-81.

24. Шимкович В. М., Кравець П. І. Моделі та методи синтезу апаратно-програмних компонентів нейромережевих систем керування // 21-ї Міжнародна конференція з автоматичного керування «АВТОМАТИКА – 2014» НТУУ «КПІ»: матеріали наук.-практ. конф. з міжнар. участю (м. Київ, 23 вересня 2014р.). Київ, 2014р. с. 170-171.

25. Шимкович В. М., Кравець П. І., Зубенко Г. А. Моделі штучних нейронних мереж при їх апаратно-програмній реалізації на FPGA // XIV международная научная конференция им. Т.А.Таран «Интеллектуальный анализ информации ИАИ-2014»: сборник трудов (г. Киев, 14-16 мая 2014г.) г. Киев, 2014. с. 4-11.

26. Шимкович В. Н., Кравец П. И., Лукина Т. И., Жеребко В. А. Двухэтапная оптимизация в многообъектных иерархических системах управления на базе генетических алгоритмов // XIII международной научной конферен-

ции имени Т. А. Таран «Интеллектуальный анализ информации ИАИ-2013»: сборник трудов (г. Киев, 15-17 мая 2013 г.). г. Киев, 2013. с. 248-254.

27. Шимкович В. М., Кравець П. І., Ткач І. І. Розробка технології оцінювання показників нейромережевих моделей об'єктів керування // Обчислювальний інтелект: матеріали наук.-практ. конф. з міжнар. участю (м. Черкаси, 14-18 травня 2013р.). м. Черкаси, 2013р. с. 201-202.

28. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Програмні засоби реалізації оптимізаційних алгоритмів керування складними технічними системами та комплексами // Системний аналіз та інформаційні технології: матеріали 15-ї міжнародної наук.-техн. конф. (м. Київ, 27-31 травень 2013р.) м. Київ, 2013р. с. 291-292.

29. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Концепція єдиного підходу до вирішення оптимізаційних задач в ієрархічних технічних системах керування // Системний аналіз та інформаційні технології: матеріали 14-ї міжнародної наук.-техн. конф. (м. Київ, 24 квітня 2012 р.). м. Київ, 2012р. с. 80-81.

30. Шимкович В. М., Кравець П. І. Синтез нейромережевих регуляторів систем керування складними динамічними об'єктами // Контроль і управління в складних системах: матеріали XI-ї міжнародної наук.-практ. конф. (м. Вінниця, 19-21 жовтня 2012р.). м. Вінниця, 2012р. с. 251-252.

31. Шимкович В. М., Кравець П. І., Романенко В. О., Ткач А. Б. Нейромережева система керування складним динамічним об'єктом на основі оберненої моделі // Автоматика / Automatics – 2011: матеріали XVIII міжнародної конференції з автоматичного (м. Львів, 28-30 вересня 2011р.). м. Львів, 2011. с. 312-313.

32. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Енергозберігаючі алгоритми оптимального керування багатооб'єктними розподіленими технічними комплексами // Системний аналіз та інформаційні технології: матеріали міжнар. наук.-практ. конф. (м. Київ, 23 травня 2011р.). м. Київ, 2011р. с. 270-271.

33. Шимкович В. М., Кравець П. І. Вирішення задачі оптимального та енергозберігаючого керування в багатооб'єктних розподілених технічних комплексах за допомогою інтелектуальних технологій // Обчислювальний інтелект: матеріали міжнар. наук.-практ. конф. (м. Черкаси, 10 травня 2011р.) м. Черкаси, 2011р. с. 190-191.

34. Шимкович В. М., Кравець П. І., Жеребко В. А., Дьомін Р.Ю., Мостович А. В. Нейромережеві технології оперативного діагностування технічного стану рухомого складу // Вагони нового покоління: із XX в XXI сторіччя: матеріали 73 міжнародної наук.-практ. конф. (м. Харків, 12 квітня 2011р.). м. Харків, 2011р. с. 105-106.

35. Шимкович В. М., Кравець П. І., Юрчук Л. Ю., Жеребко В. А. Інтелектуальна медична система для формування інформаційного портрету протікання хвороби // Біомедична інженерія і технологія: матеріали II-ї міжнар. наук.-практ. конф. (м. Київ, 17-18 березня 2011р.). м. Київ, 2011р. с. 250-251.

36. Шимкович В. М., Кравець П. І., Жеребко В. А. Інтелектуальні засоби діагностики і прогнозування стану організму // Біомедична інженерія і технологія: матеріали II-ї міжнар. наук.-практ. конф. (м. Київ, 17-18 березня 2011р.). м. Київ, 2011р. с. 98-99.

37. Шимкович В. М., Кравець П. І., Жеребко В. А. Методика апаратно-програмної реалізації однонейронного нейромережевого ПІД-регулятора на FPGA // Контроль і управління в складних системах: матеріали X-ї міжнародної наук.-практ. конф. (м. Вінниця, 19-21 жовтня 2010 р.). м. Вінниця, 2010 р. с. 321-322.

38. Шимкович В. М., Кравець П. І., Жеребко В. А. Програмні засоби реалізації оптимального керування в складних технічних комплексах // Розподілені комп’ютерні системи. Проектування, обчислення, застосування: матеріали ювілейної міжнародної наук.-практ. конф. (м. Київ, 6-8 квітня 2010 р.). м. Київ, 2010 р. с. 107-108.

Всього за тематикою дисертації:

За тематикою дисертаційної роботи опубліковано 38 наукових робіт.

Зокрема, опубліковано 10 статей в наукових фахових журналах України; 3 наукових статей у закордонних виданнях, проіндексованих у базі даних Scopus, з яких 2 статті в журналах 3-го квартиля (Q3) та 1 стаття в журналі 4-го квартиля (Q4).

Крім того, результати роботи відображено у 2 статтях в інших наукових журналах України та за кордоном, а також у 23 публікаціях у матеріалах науково-технічних конференцій, з яких 3 видань проіндексовано у базі даних Scopus.

Total for the dissertation theses:

On the topic of the dissertation, 38 scientific publications have been published. The main scientific results are shown in 20 scientific publications.

In particular, the following scientific publications have been published: 10 articles in the professional scientific journals of Ukraine; 3 articles in professional foreign journals indexed in the reference base Scopus, where 2 articles are in the journals of 3-rd quartile (Q3) and 1 article is in the journals of 4-th quartile (Q4).

Besides, the results are presented in 2 articles in other scientific journals of Ukraine and abroad as well as in 23 proceedings of scientific conferences, including 3 proceedings indexed in the reference base Scopus.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

ВР – Back-Propagation (алгоритм навчання зворотного розповсюдження помилки

DSP – цифрові сигнальні процесори

LUT – look-up table (логічна таблиця) FPGA

FPGA – програмовані логічні інтегральні схеми (структури)

ПЛМ – програмовані логічні матриці

ОЗП – оперативний запам'ятовуючий пристрій

ЗВІС – замовлена до виготовлення інтегральна схема

ШНМ – штучна нейронна мережа

БНМ – багатошарові нейронні мережі

РБФ – радіально-базисні функції

НМС – нейромережеві структури

НМА – нейромережеві алгоритми

НСК – нейромережеві системи керування

НК – нейроконтролер

НЕ – нейроемулятор

ОК – об'єкт керування

СК – система керування

УНО – узагальнений налагоджувальний об'єкт

ВСТУП

Актуальність роботи. Сучасний етап розвитку науки та техніки характеризується швидким зростанням складності створюваних технічних систем. Керування такими системами вимагає розробки нових методів керування, оскільки модифікація та вдосконалення традиційних прийомів керування не завжди забезпечує виконання жорстких вимог до показників якості керування. Класичні методи керування в основному спираються на теорію лінійних систем, в той час як більшість реальних об'єктів є нелінійними. Проблема синтезу систем керування в умовах невизначеності є на даний час однією з центральних в сучасній теорії автоматичного керування. Складність вирішення цієї проблеми обумовлена складністю самого об’єкта керування, структурними, параметричними та інформаційними невизначеностями в описі об'єкта керування, та складністю задач керування, багатокритеріальністю оптимізаційних задач, відсутністю можливих аналітичних рішень, необхідності врахування всіх властивостей збурень та інше. Вирішення цієї проблеми вимагає пошуку альтернативних підходів до проєктування систем керування, одним з яких передбачає впровадження нейромережевих систем.

Нейромережеві системи керування є високотехнологічним напрямком теорії керування та відносяться до класу нелінійних динамічних систем. Висока швидкодія за рахунок розпаралелювання вхідної інформації в поєднанні зі здатністю до навчання нейронних мереж робить цю технологію вельми привабливою для створення пристроїв керування в автоматичних системах. Нейронні мережі можуть бути використані для побудови регулюючих та коректуючих пристроїв, еталонних, адаптивних, номінальних та інверсно-динамічних моделей об'єктів, на основі яких здійснюється дослідження об’єктів, аналіз впливу збурень, що діють на об’єкт, визначення оптимального закону керування, пошук або обчислення оптимальної програми зміни керуючого впливу при зміні значень параметрів об’єкту та характеристик вхідних даних. Кім того нейронні мережі можуть бути використані для ідентифі-

кації об'єктів, прогнозування стану об'єктів, розпізнавання, кластерізації, класифікації, аналізу великої кількості даних, що надходять з високою швидкістю з великої кількості пристроїв та датчиків, тощо. Здатність до навчання на заданий принцип функціонування дозволяє створювати системи автоматичного керування, оптимальні по швидкодії, енергоспоживанню та ін. Природно, що при цьому можлива реалізація декількох принципів функціонування та перехід з одного на інший. Вони є універсальним засобом для моделювання багатовимірних нелінійних об'єктів та знаходження рішень в некоректних задачах.

Фундаментальні наукові результати в галузі розробки інтелектуальних та нейромережевих систем відображені в роботах вітчизняних і зарубіжних вчених: Вербоса П., Зайченка Ю.П., Кохонена Т.К., Пайперта С., Розенблата Ф., Сарідіса Дж., Стиренка С.Г., Сигеру Омату, Терехова В.А., Терейковського І.А., Федосова Е.А., Федунова Б.Є., Хопфілда Д.Д., Широчина В.П. та ін.

На даний час основним методом реалізації нейромережевих систем керування є програмний, за допомогою архітектури Фон Неймана, яка є послідовною, всупереч притаманному паралелізму ШНМ. А також програмна реалізація за допомогою графічних процесорів – GPU, що дозволила підвищити ефективність програм з ШНМ в порівнянні з архітектурою Фон Неймана. Програмна реалізація систем керування значно звужує коло їх практичного застосування через значну вартість, що робить недоцільним їх використання в вбудованих системах керування технічними об’єктами. Крім того, програмна реалізація має низку швидкодію та потребує значних затрат часу на навчання через необхідність виконання великої кількості послідовних обчислень. Для розв’язання більшості задач, які вирішуються в вбудованих системах керування технічними об’єктами, вимагається висока швидкодія мережі та її навчання. Рекурентність і послідовність дій процедури навчання нейромережі, при її реалізації на всій множині налагоджуваних параметрів, не дозволяє повністю вирішити проблему швидкодії нейронної мережі та процедури навчання в реальному часі. Єдиною альтернативою цьому є розпаралe-

лення процедури навчання та функціонування внутрішніх елементів нейромережі. Такі можливості з'являються при реалізації ШНМ на програмованих інтегральних логічних схемах(FPGA).

Таким чином є актуальна науково-технічна задача реалізації нейромережевих компонентів систем керування динамічними об'єктами, що забезпечують адаптацію та самоналагодження систем керування в реальному часі. Для досягнення цього необхідна розробка нових методів та алгоритмів реалізації апаратних компонентів нейромережевих систем керування.

Зв’язок роботи з науковими програмами, планами, темами. Дослідження, представлені у дисертації, проводились в рамках держбюджетних науково-дослідних робіт Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського»: «Розробка методів та засобів проектування і реалізації оптимального та енергозберігаючого керування в багатооб'єктних розподілених технічних комплексах» (номер державної реєстрації 0110U002196); «Розробка методів і засобів апаратно-програмної реалізації нейроконтролерів на основі програмованих логічних інтегральних схем для побудови інтелектуальних систем керування» (номер державної реєстрації 0113U000223); «Платформа розроблення, експлуатації і розвитку критичних IT-інфраструктур для роботи з великими даними» (номер державної реєстрації 0116U003801); «Розробка та впровадження системи керування IT-інфраструктурою з консолідованими інформаційно-обчислювальними ресурсами» (номер державної реєстрації 0115U000322); «Хмарна платформа розроблення і керування функціонуванням критичних IT-інфраструктур, що опрацьовують великі обсяги даних» (номер державної реєстрації 0117U000537); Дослідно-конструкторська робота «Розробка документації для апаратно-програмного комплексу CashBag System» (реєстраційний номер 2/577-18).

Мета і завдання дослідження. Метою дисертаційної роботи є підвищення ефективності нейромережевих систем керування за рахунок створення

швидкодіючих компонентів, які дозволяють реалізовувати функції ідентифікації, адаптації та керування об'єктами в реальному часі.

Задачі дослідження.

1. Провести аналіз існуючих нейромережевих систем керування і основних типів нейронних мереж, які при цьому використовуються, методів їх адаптації та методів, засобів, технологій та алгоритмів їх апаратної реалізації.

2. Розробити метод та алгоритми апаратної реалізації нейронних мереж на багатопотокових обчислювальних системах, для подальшого синтезу на їх базі компонентів нейромережевих систем керування.

3. Розробити метод проєктування апаратних компонентів нейромережевих систем керування.

4. Розробити метод проєктування апаратних компонентів контуру адаптації нейромережевих систем керування.

5. Побудувати та дослідити основні типи нейромережевих систем керування динамічними об'єктами на основі розроблених компонентів.

6. Розробити діючий макет нейромережевого контролера та впровадити розроблені технології, методи, засоби та алгоритми в системі керування динамічним об'єктом.

Об’єкт дослідження – нейромережеві системи керування динамічними об'єктами.

Предмет дослідження – методи та технології автоматизованого проєктування інтелектуалізованих апаратних засобів нейромережевих систем керування динамічними об'єктами.

Методи дослідження. Методологічну основу дослідження становлять фундаментальні положення сучасної теорії автоматичного керування, наукові дослідження вітчизняних і зарубіжних вчених у сфері нейромережевих систем керування. Аналіз паралельних процесів та похибки обчислень проводиться з застосуванням елементів математичного аналізу, теорії численних методів, імітаційного моделювання.

Наукова новизна одержаних результатів.

1. *Уперше* розроблено метод проєктування нелінійних функцій активації штучного нейрону на програмованих логічних інтегральних схемах, який відрізняється від існуючих методів проєктування тим, що коефіцієнти шматково-лінійної апроксимації функції активації зберігаються у пам'яті тільки для позитивних або тільки для негативних значень аргументу, що дозволило оптимізувати кількість використаного обчислювального ресурсу та збільшити швидкодію обчислень нейронної мережі;

2. *Уперше* розроблено метод проєктування апаратних компонентів нейромережевих систем, таких як пряма та інверсна модель об'єкта керування, на програмованих логічних інтегральних схемах, який відрізніться використанням розроблених методів та засобів, що дозволяє підвищити рівень автоматизації проєктування нейромережевих моделей при їх апаратній реалізації;

3. *Уперше* розроблено метод проєктування апаратного компоненту оптимізації вагових коефіцієнтів нейронних мереж за допомогою генетичного алгоритму при реалізації його на програмованих логічних інтегральних схемах, який відрізняється від існуючих реалізацією операцій мутації та кросовера, що дозволяє значно підвищити швидкість оптимізації вагових коефіцієнтів нейронних мереж;

4. *Дістав подальший розвиток* комплексний формалізований підхід до реалізації багатоетапної процедури ідентифікації складних динамічних об'єктів з використанням нейронних мереж, що відрізняється від існуючих етапом створення сукупності нейромережевих моделей та їх оцінювання, що дозволяє обрати мінімальну структуру нейромережевої моделі для подальшої реалізації.

Практичне значення отриманих результатів. Отримані в дисертаційній роботі результати можуть бути використані для проєктування контролерів нейромережевих систем керування здатних функціонувати і адаптуватися в реальному часі.

Практична цінність роботи полягає в наступному:

– розроблено алгоритми апаратної реалізації штучних нейронних мереж прямого розповсюдження та RBF-мереж на основі програмованих логічних інтегральних схем, що дозволяють підвищити швидкодію таких апаратних блоків, зменшити кількість обчислювального ресурсу необхідного для їх реалізації та підвищити точність;

– розроблено програмний пакет для дослідження та оцінки нейромережевих моделей багатовимірних об'єктів керування для їх подальшої реалізації;

– розроблено узагальнену структурну схему нейроконтролера на основі програмованих логічних інтегральних схем, що реалізує базові компоненти нейромережевих систем керування;

– реалізовано макет нейроконтроллера для системи стабілізації рухомого об'єкта на обмеженій площині та проведено його дослідження, які підтвердили високу ефективність роботи;

Теоретичні та практичні результати роботи впроваджено у ТОВ «СІТІУС ПРО» (акт про впровадження від 20.07.2020) та ТОВ «АЙАДМІН» (акт про впровадження від 12.08.2020).

Результати роботи впровадженні у навчальний процес кафедри автоматики та управління в технічних системах Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського», починаючи з 2013-2014 навчального року в матеріалах лекцій з навчальних дисциплін «Теорія штучного інтелекту в управлінні», в якій введено новий розділ «Технологія апаратно-програмної реалізації нейромережевих систем», та «Проєктування комп'ютеризованих систем керування», а також при проведенні курсового та дипломного проєктування.

Особистий внесок здобувача. Всі основні результати дисертаційного дослідження, які представлені до захисту, одержані автором особисто.

У працях, опублікованих у співавторстві, особистий внесок здобувача є наступним: [1 – 3, 5, 7, 16, 18, 20, 31] – узагальнена структурна схема нейро-

мережевого контролера, моделі системи стабілізації рухомого об'єкта, реалізація та дослідження нейромережевих контролерів; [4, 12] – методика апаратної реалізації адаптивного ПІД-регулятора на одному нейроні та його дослідження; [6, 11, 19, 25] – метод та алгоритми апаратної реалізації штучних нейронних мереж; [8] – метод оптимізації вагових коефіцієнтів нейронної мережі за допомогою генетичного алгоритму при реалізації на ПЛІС; [9, 15, 17, 20 – 24, 30, 31] – метод синтезу нейромережевих компонентів систем керування динамічними об'єктами; [10, 27] – модифікований комплексний підхід до реалізації багатоетапної процедури ідентифікації об'єктів керування за допомогою нейронних мереж; [13, 34] – нейромережева модель технології оперативного діагностування технічного стану рухомого складу; [14, 37] – методика апаратної реалізації ПІД-регулятора на одному нейроні та його дослідження; [28, 29, 32, 33, 38] – алгоритм оптимізації в багатооб'єктних ієрархічних системах керування на базі генетичних алгоритмів; [35, 36] – реалізація та дослідження інтелектуальних моделей.

Апробація результатів дисертації. Основні положення та результати проведених у дисертаційній роботі досліджень доповідались та обговорювались на 21 міжнародній та всеукраїнській науково-технічній конференції: Ювілейна міжнародна науково-практична конференція «Розподілені комп'ютерні системи» (Київ, НТУУ «КПІ», 2010 р.); X-та міжнародна науково-практична конференція «Контроль та керування складними системами» (Вінниця, ВНТУ, 2010р., 2012 р.); 73 міжнародна конференція «Вагони нового покоління: із XX в XXI сторіччя» (Харків, 2011); Міжнародна конференція «Системний аналіз та інформаційні технології» (Київ, НТУУ «КПІ», 2011, 2012, 2013); Міжнародна конференція «Обчислювальний інтелект – 2011» (Черкаси, ЧДТУ, 2011р.); Міжнародна конференція з автоматичного керування «АВТОМАТИКА/ AUTOMATICS – 2011» (Львів, НУ «Львівська політехніка», 2011); XIII та XIV міжнародній науковій конференціях імені Т.А. Таран «Інтелектуальний аналіз інформації» (Київ, НТУУ «КПІ», 2013, 2014); XXI міжнародна конференція з автоматичного керування «АВТОМАТИКА/

AUTOMATICS – 2014» (Київ, НТУУ «КПІ», 2014); І Міжнародна конференція присвячена 70-річчю кафедри автоматики та керування в технічних системах, м. Київ, 24-25 листопада 2015 р.; Congress on Information Technology, Computational and Experimental Physics December 18-20, 2015 Cracow, Poland. 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Bucharest, Romania, 2017; The First International Conference on Computer Science, Engineering and Education Applications (ICCSEEA2018) 18-20 January 2018, Kiev, Ukraine; Одинадцята міжнародна науково-практична конференція з програмування УкрПРОГ'2018, Україна, Київ, 22-24 травня 2018 р.; The Second International Conference on Computer Science, Engineering and Education Applications (ICCSEEA2019) 26-27 January 2019, Kiev, Ukraine; 2020 IEEE 7th INTERNATIONAL CONFERENCE ON ENERGY SMART SYSTEMS, April 23-25, 2020 Kyiv, Ukraine.

Публікації. Результати досліджень за тематикою дисертації опубліковано у 38 наукових працях, з яких 15 наукових статей (з них 4 статті у періодичних наукових виданнях інших держав, які входять до ОЕСР та/або Європейського Союзу, з яких 3 наукові статті у проіндексовані у наукометричній базі Scopus, в тому числі 2 статті в журналах 3-го квартіля (Q3) та 1 стаття в журналі 4-го квартіля (Q4), 10 статей у наукових фахових виданнях України та 1 стаття в інших наукових виданнях України) та 23 публікації в матеріалах наукових та науково-технічних конференцій (в тому числі 3 публікації в матеріалах наукових конференцій, що реферуються наукометричними базами Scopus та/або Web of Science, і у 20 матеріалах інших наукових конференцій в Україні та за кордоном).

Структура дисертації. Дисертація складається зі вступу, чотирьох розділів, висновків, списку літературних джерел, додатків. Робота містить 131 сторінку основного тексту, 10 таблиць, 66 рисунків, 158 використаних джерел, 2 додатки. Загальний обсяг дисертації становить 177 сторінок.

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ ТА ЗАСОБІВ ПОБУДОВИ НЕЙРОМЕРЕЖЕВИХ СИСТЕМ КЕРУВАННЯ ДИНАМІЧНИМИ ОБ'ЄКТАМИ

1.1 Аналіз принципів побудови нейромережевих систем керування динамічними об'єктами

В історії розвитку теорії автоматичного керування чітко виділяються три етапи [1, 2], перший етап класичної детермінованої теорії автоматичного керування, що охопила період часу з кінця XIX по 40-і роки XX століття. У цей період основними завданнями керування були задача стійкості та задача про якість перехідних процесів. Другий етап теорії керування почався в 40-50-х роках нашого століття і тривав приблизно до середини 70-х років. Це – етап класичної схоластичної теорії автоматичного керування. Він характеризується новою постановкою основного завдання теорії керування: врахувати випадкові збурення, що діють на систему і забезпечити якісну роботу в умовах постійно діючих перешкод. Близько 25-ти років назад у розвитку теорії автоматичного керування почався новий етап, пов'язаний з адаптивною постановкою основного завдання керування, особливість полягає у відсутності початкових знань про математичну модель об'єкта керування, будь то диференціальні рівняння або щільності ймовірностей випадкових зовнішніх впливів. Об'єкт – це чорний ящик, що піддається невідомим випадковим впливам, доступні лише його входи і виходи.

Динамічна система (*об'єкт керування*) – математична абстракція, призначена для опису і вивчення систем, еволюція в часі яких однозначно визначається початковим станом.

В адаптивній постановці об'єкт керування описується своєю функціональною моделлю:

$$P\{\mathbf{u}(t),\mathbf{y}(t)\}, \tag{1.1}$$

яка зв'язує вектор вхідних впливів $\mathbf{u}(t)$ з вектором вихідних сигналів $\mathbf{y}(t)$.

На Рис. 1.1 представлена динамічна система (об'єкт керування). Такий опис бере свій початок від ідеї «чорного ящика» і не є портретом динамічної поведінки об'єкта, а відображає тільки його функціональні зв'язки.

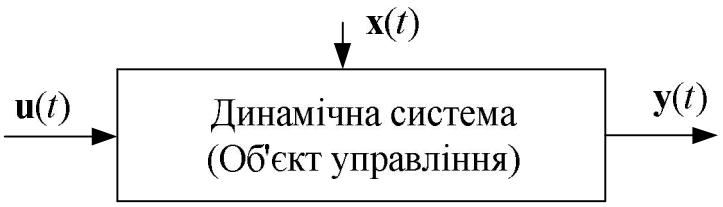


Рис. 1.1 – Динамічна система

Змінні $\mathbf{u}(t) = (u_1(t), u_2(t), ..., u_p(t))^T$, $\mathbf{x}(t) = (x_1(t), x_2(t), ..., x_n(t))^T$, і $\mathbf{y}(t) = (y_1(t), y_2(t), ..., y_m(t))^T$ є векторами. Тут $\mathbf{u}(t)$ і $\mathbf{y}(t)$ – вектори входу і виходу системи, відповідно, $\mathbf{x}(t)$ – змінна стану системи, p – розмірність вхідного простору, m – розмірність вхідного простору, а n – порядок системи.

Класично, динаміка такої системи описується системою диференціальних рівнянь [2]:

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= \Phi(\mathbf{x}(t), \mathbf{u}(t)), \quad t \in \Re^+; \\ \mathbf{y}(t) &= F(\mathbf{x}(t)). \end{aligned} \tag{1.2}$$

Тут вектори функцій $\Phi = (f_1, f_2, \ldots, f_n)$ і $F = (f_1, f_2, \ldots, f_m)$ – статичні нелінійні перетворення. $\Phi : \Re^n \times \Re^p \to \Re^n$, $F : \Re^n \to \Re^m$. Вектор $\mathbf{x}(t)$ описує стан системи в момент часу t. Він визначається станом системи в початковий момент $t_0<t$ і виходом $\mathbf{u}(t)$, визначеному на інтервалі $[t_0, t)$. Вихід системи $\mathbf{y}(t)$ повністю визначається станом системи $\mathbf{x}(t)$ в момент t.

Іншим підходом є дискретний опис динамічної системи. При розбитті за часом $t_0, t_1, t_2, ...$, де $t_{i+1}=t_i+\Delta t$, і позначенні $\mathbf{x}(t_k)$, $\mathbf{y}(t_k)$ і $\mathbf{u}(t_k)$ як $\mathbf{x}(k)$, $\mathbf{y}(k)$ і $\mathbf{u}(k)$ відповідно. Тоді динаміку системи можна описати наступними різницевими рівняннями:

$$\begin{aligned} \mathbf{x}(k+1) &= \Phi(\mathbf{x}(k), \mathbf{u}(k)); \\ \mathbf{y}(k) &= F(\mathbf{x}(k)). \end{aligned} \tag{1.3}$$

Тут Ф і *F* аналогічні перетворенням в (1.2).

Рівняння (1.2) та (1.3) представляють динаміку системи, як перетворення вхід-вихід [2]. Для широкого класу задач ці форми подання рівнозначні і можуть бути зведені один до одного. Мета системи керування (СК), таким об'єктом, полягає в тому, щоб в процесі функціонування визначити закон регулювання, що забезпечує задану поведінку об'єкта. Для вирішення цього завдання на додаток до основного контуру в систему керування вводиться контур адаптації [3, 4].

Управляюча система, що автоматично визначає потрібний закон керування за допомогою аналізу поведінки об'єкта при поточному управлінні, називається адаптивною [4].

Адаптивні системи можна розділити на два великі класи: *самоорганізуючі* та *самоналагоджувальні* [3, 4].

У самоорганізуючих системах в процесі функціонування відбувається формування алгоритму керування (його структури і параметрів), що дозволяє оптимізувати систему з точки зору поставленої мети керування. Такі завдання виникають, наприклад, в умовах зміни структури і параметрів об'єкта керування залежно від режиму функціонування, коли апріорної інформації недостатньо для визначення поточного режиму функціонування об’єкта. При широкому класі можливих структур об'єкта важко сподіватися на вибір єдиної структури алгоритму керування, здатної забезпечити замкнутій системі досягнення мети керування в всіх режимах функціонування. Тому, мова йде про синтез при вільній структурі регулятора. Очевидна складність постановки завдання не дозволяє надіятися на прості алгоритми її вирішення, а отже, і на широке впровадження таких систем в практику [3, 4].

Завдання істотно спрощується, якщо структура об'єкта керування відома і незмінна, а поведінка залежить від ряду невідомих параметрів. Це завдання вирішується в класі самоналагоджувальних систем (СНС), в яких структура регулятора задана і потрібно визначити лише алгоритм налаштування коефіцієнтів регулятора.

СНС діляться на два підкласи [4]: пошукові та без пошукові. У пошукових СНС мінімум (або максимум) міри якості (продуктивність установки, витрата палива і т.д.) визначається за допомогою спеціально організованих пошукових сигналів. Найпростішими пошуковими системами є більшість екстремальних систем, в яких недолік апріорної інформації заповнюється за рахунок поточної інформації, одержуваної у вигляді реакції об'єкта на пошукові впливи.

У безпошукових СНС в явному або неявному вигляді є модель з бажаними динамічними характеристиками [4]. Завдання алгоритму адаптації полягає в налаштуванні коефіцієнтів регулятора таким чином, щоб звести неузгодженість між об'єктом керування і моделлю до нуля. Таке керування називають прямим адаптивним керуванням, а системи – адаптивними системами з еталонною моделлю. У разі непрямого адаптивного керування спочатку проводять ідентифікацію об'єкта, а потім визначають відповідні коефіцієнти регулятора. Такі регулятори називаються регулятори з самоналаштуванням.

При прямому адаптивному управлінні [4] контури адаптації працюють по замкнутому циклу. Це дозволяє парирувати зміни параметрів об'єкта і регулятора в процесі функціонування. Проте кожен контур самонастроювання підвищує порядок системи як мінімум на одиницю, і при цьому істотно впливає на загальну динаміку замкнутої системи.

У разі непрямого адаптивного керування [4] контури самонастроювання працюють по розімкненому циклу і не впливають на динаміку системи. Проте всі помилки ідентифікації, відхилення параметрів об'єкта і регулятора істотно впливають на точність керування.

У безпошукових самоналагоджувальних системах [4] еталонна модель може бути реалізована у вигляді реальної динамічної ланки (явна модель) або бути присутньою у вигляді деякого еталонного рівняння, що зв'язує регульовані змінні та їх похідні (неявна модель). У неявній моделі коефіцієнти еталонного рівняння є параметрами алгоритму адаптації.

З самого початку третього етапу, пов'язаного з адаптивною постановкою основного завдання керування, з'явилася величезна кількість наукових статей і доповідей[5 – 13], присвячених розробкам у галузі ШНМ у контексті адаптивної теорії керування, а саме реалізації на ШНМ таких компонентів адаптивних систем як регулятори та моделі об'єкта [6–8]. Застосування ШНМ для реалізації СУ має багато переваг, завдяки таким властивостям: ШНМ є ідеальними апроксиматорами будь-якої нелінійної функції багатьох змінних, що дає можливість моделювати ОК та формувати функції керування будь-якої складності; внутрішня адаптивність завдяки можливості самонавчання; висока швидкодія та паралельна обробка інформації.

Нейронні мережі можуть бути використані в таких об’єктах як робототехніка, керування безпілотними літальними апаратами, керуванні транспортними засобами, розпізнаванні образів, аналізі та прийнятті рішень в системах Інтернету Речей, керуванні космічними кораблями, військовою технікою та багатьма іншими різними сферами застосування в сучасних технологіях [14 – 20], на Рис.1 представлено сучасні технічні засоби,що використовують додатки з ШНМ. У цих системах нейронні мережі можуть використовуватися для ідентифікації об'єктів, прогнозування стану об’єктів, розпізнавання, кластеризації, класифікації, аналізу великої кількості даних, що надходять з великою швидкістю від великої кількості пристроїв та датчиків тощо [21 – 33].

При описі систем керування з нейромережевими елементами та реалізації їх на сучасній елементній базі більш зручним є дискретне представлення ОК, тому надалі будемо розглядати динаміку системи у вигляді (1.3).

Властивості ШНМ дозволяють моделювати складні нелінійні динамічні об'єкти керування – у вигляді *прямих* та *інверсних* моделей по вимірах «вхід-вихід» цього об'єкта [18 – 22, 34]. Обидві моделі використаються для обчислення векторів стану об'єкта і формування функції керування ним. Вони можуть бути базовими для структурного синтезу функціонально більш складних систем керування.

Рис. 1.2 – Сучасні технічні засоби, що використовують додатки з ШНМ

Пряму модель ОК по вимірах вхідних $u(k)$ та вихідних $y(k)$ даних можна отримати по схемі (рисунок 1.3*а*), відомій в теорії ідентифікації, як схема з налагоджуваною моделлю [23, 24, 34]. Вона реалізується в даному випадку рекурентною штучною мережею, що навчається на основі помилки $\hat{e}(t)$:

$$\hat{y}(k+1) = \mathbf{F}(u_k, z^{-1}u_k, ..., z^{-m}u_k;\ \hat{y}_k, z^{-1}\hat{y}_k,\ z^{-n}\hat{y}_k;\ \mathbf{w}_i^{(l)}).$$

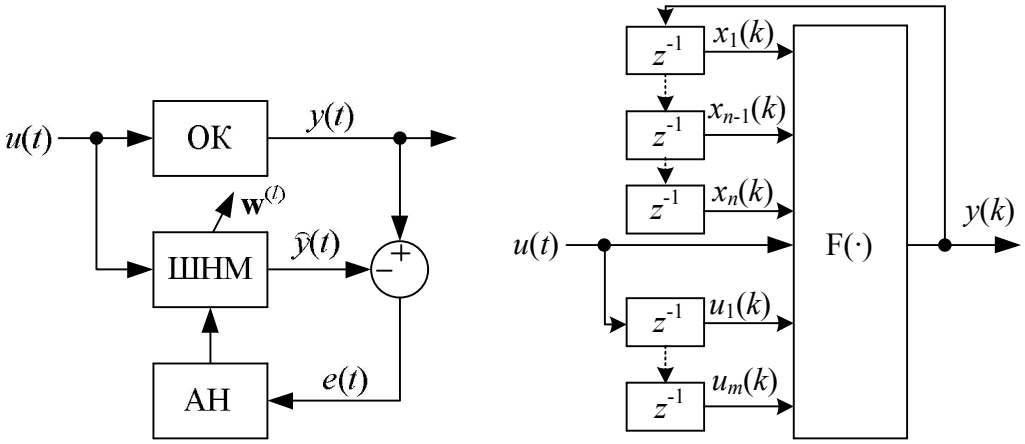
Вектором стану мережі обрано $col(u, z^{-1}y, ..., z^{-n}y) = col(\hat{x}_n(k), \hat{x}_{n-1}(k), ..., \hat{x}_1(k))$, де z^{-1} – оператор затримки. Результатом ідентифікації динамічної моделі реального ОК, який в загальному випадку може бути представлений параметрично недовизначеним нелінійним диференціальним рівнянням

$$y(k+1) = f[y(k), y(k-1), ..., y(k-n+1); u(k), ..., u(k-m+1)], \qquad (1.4)$$

є налаштовані значення вагових коефіцієнтів $\mathbf{w}_i^{(l)}$ у шарах $l = \overline{1, K}$ та оцінки вектора стану об'єкта (1.4), одержаних в процесі навчання ШНМ шляхом мі-

німізації помилки $\hat{e}(t) = y(t) - \hat{y}(t)$ або мінімуму деякого функціонала $J\left(\hat{e}(t)\right)$.

Схема динамічної нейромережі з одним входом й одним виходом, що відповідає схемі прямої моделі навчання з використанням вимірів «вхід-вихід» ОК зображена на Рис. 1.3*б*.



а) *б*)

Рис. 1.3 – Схеми ідентифікації об’єкта керування (*а*) та прямої моделі об'єкта керування (*б*) у вигляді ШНМ

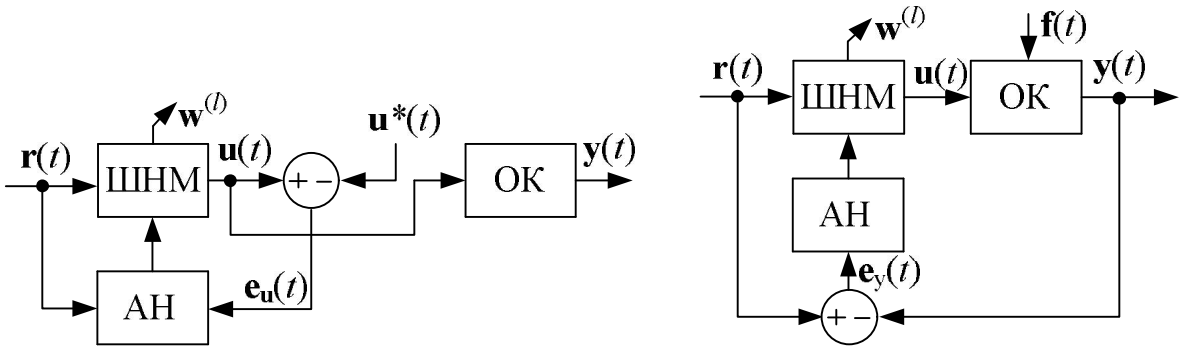
Засновані на ШНМ ідентифікаційні моделі називаються нейроемуляторами (НЕ). У загальному вигляді вони описуються нелінійним рівнянням (1.4).

Для одержання прямої моделі в заданому класі функцій $\mathbf{u}(t) \in \mathbf{U}$ не потрібно повної апріорної інформації про структуру зв'язків й їхніх операторів для ОК, окрім інформації про стійкість та обмеженість всіх траєкторій $\mathbf{y}(t)$ для $t \geq 0$. Варто зауважити, що навчена мережа враховує вплив на реальний ОК зовнішніх збурень.

Для побудови інверсної, нейромережевої моделі ОК, нейроконтроллера (НК), можуть бути використані два підходи навчання ШНМ: узагальнене та спеціалізоване.

В схемі узагальненого інверсного навчання, Рис. 1.4*а*, в якості вхідних даних використається тестовий сигнал $\mathbf{u}^*(t)$. Це може бути, наприклад, значення відомої функції оптимального управляння об'єктом. Вихід реального

ОК $\mathbf{y}(t)$ при підключенні виходу навченої мережі до входу об'єкта відтворює значення функції $\mathbf{r}(t)$, тобто якщо A – оператор ОК, то при $\mathbf{e}_u(t)=0$ нейромережа відображає інверсний оператор A^{-1}, що й пояснює термін «інверсне навчання». Тестовий сигнал $\mathbf{u}^*(t)$ повинен задовольняти вимозі повного перекриття можливого діапазону зміни керуючого впливу в реальній СК.



а) *б*)

Рис. 1.4 – Схеми інверсного навчання ШНМ: *а* – загального; *б* – спеціалізованого

Другий підхід до інверсного моделювання ОК реалізується схемою спеціалізованого інверсного навчання. Ця ж схема дозволяє відтворювати задану функцію $\mathbf{r}(t)$, рисунок 1.4*б*. У даній схемі сигнал $\mathbf{r}(t)$ виконує роль тестового в завданні інверсного моделювання ОК за допомогою нейромережі і його форма відповідає класу відтворених у завданні керування функцій.

На відміну від узагальненої схеми в спеціалізованій використовується помилка між заданою функцією $\mathbf{r}(t)$ і виходом $\mathbf{y}(t)$, мережа налагоджується за динамічним алгоритмом навчання. Якщо нейромережа навчена і помилка $\mathbf{e}_y(t)=0$, то функція $\mathbf{r}(t)$ у точності відтворюється на виході об'єкта. В [34] стверджується, що схема спеціалізованого інверсного навчання дозволяє відтворити точну інверсну модель ОК при використанні реального виходу $\mathbf{y}(t)$.

Інверсні моделі самі по собі можуть бути застосовні для керування динамічними об'єктами, у тому числі і для адаптивного керування [28, 29].

Системи керування на основі прямої та інверсної моделі ОК можна розділити на системи без зворотного зв'язку та системи зі зворотнім зв'язком [8, 34].

СК без зворотного зв'язку відома під назвою система нейромережевого керування послідовного типу [8, 34], структурна схема даної СК представлена Рис. 1.5.

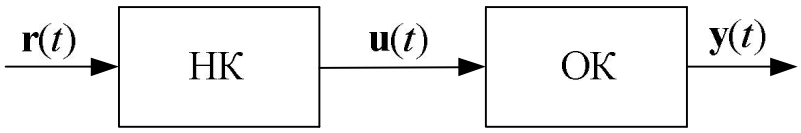


Рис. 1.5 – Схема НСК без зворотного зв'язку

В цій НСК навчена нейромережа НК виконує роль регулятора, налаштованого на обернену динаміку ОК. Схема не використовує від'ємного зворотного зв'язку, тому перевагою такої схеми є її простота та відсутність проблем зі стійкістю системи. До недоліків системи потрібно віднести наступне: при появі неконтрольованих збурень, а також не стаціонарності ОК система не гарантує, що вихід об'єкта буде відповідати заданому, схема не може забезпечити керування нестійким об'єктом, а також коли обернений оператор об'єкта дуже складний і нелінійний або взагалі не може бути реалізованим. Навчання НК в такій схемі може відбуватися в режимі попереднього навчання або в реальному часі. В режимі попереднього навчання НК навчається по схемі попереднього інверсного навчання, схема якого показана на Рис. 1.4*а*.

В схему СК без зворотного зв'язку може бути додана нейромережа-емулятор (НЕ), яка буде виконувати роль моделі об'єкта для імітаційного моделювання поведінки об'єкта з метою прогнозування його відгуку на керування, Рис. 1.6. В цій схемі НЕ може використовуватися для навчання НК або використовуватися для прогнозування виходу ОК в режимі реального часу.

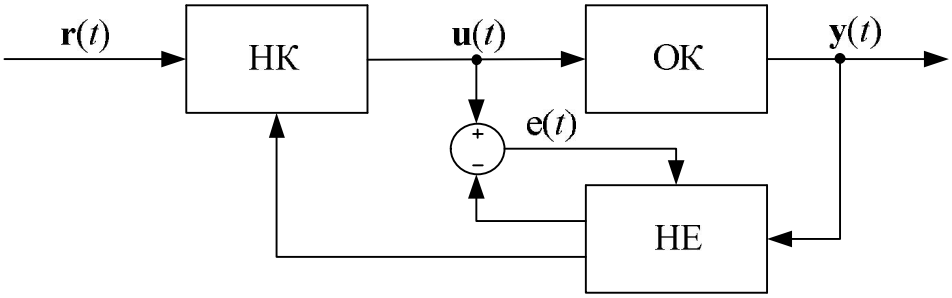


Рис. 1.6 – Схема СУ з НК та НЕ

Найпростіша структура НСК зі зворотнім зв'язком показана на рисунку 1.4б. Тут НК виконує роль регулятора замкнутої системи. Перевагою такої структури є її властивість забезпечувати високу якість регулювання в умовах неконтрольованих збурень, а також при не стаціонарності та нестійкості ОК. Однак в даному випадку налагодження НК є значно складнішим завданням: якщо для послідовної схеми відомо, що НК повинен реалізовувати обернену динаміку ОК, то в системі зі зворотнім зв'язком оператор, який повинен реалізовувати НК невідомий, зв'язок між показниками якості регулювання і параметрами НК також невідомий.

Якщо взяти до уваги, що метою адаптації СК є забезпечення якісних показників роботи системи в умовах різних видів збурень (зовнішніх та внутрішніх), то адаптивні СК можливо розглядати як один із класів оптимальних систем керування, тобто адаптацію можна ототожнити з оптимізацією в умовах недостатньої апріорної та потокової інформації [3, 4, 8, 34]. Для цього досягнення завдання (1.4) необхідно реалізовувати шляхом побудови (адаптації) закону керування, що забезпечує задане (звичайно екстремальне) значення деякого функціонала якості $I(y(t))$

$$u(t) \Rightarrow I(y(t)) \to ext\,. \qquad (1.5)$$

Узагальнена схема СК з оптимальним законом керування показана на Рис. 1.7.

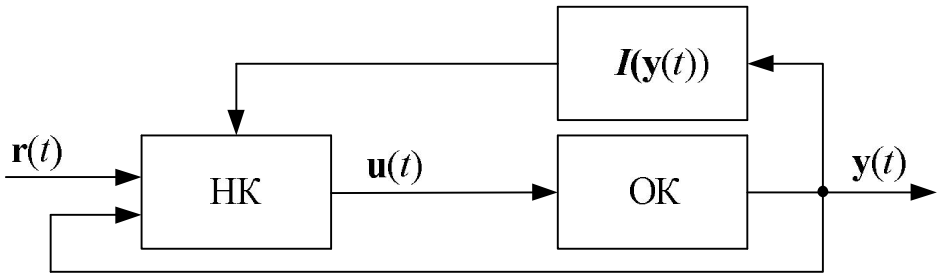


Рис. 1.7 – Схема НСК з оптимальним законом керування

У випадках, коли оптимальна функція керування динамічним об'єктом може бути аналітично обчислена та її значення визначені на кінцевому інтер-

валі часу, НСК може бути побудована за схемою інверсного адаптивного керування у вигляді узагальненого об'єкта [8, 34].

Іншим розповсюдженим типом адаптивних СК є СК з налагоджуваною моделлю ОК [8, 34], схема адаптивної СК з прямою та інверсною моделями об'єкта представлена на Рис. 1.8.

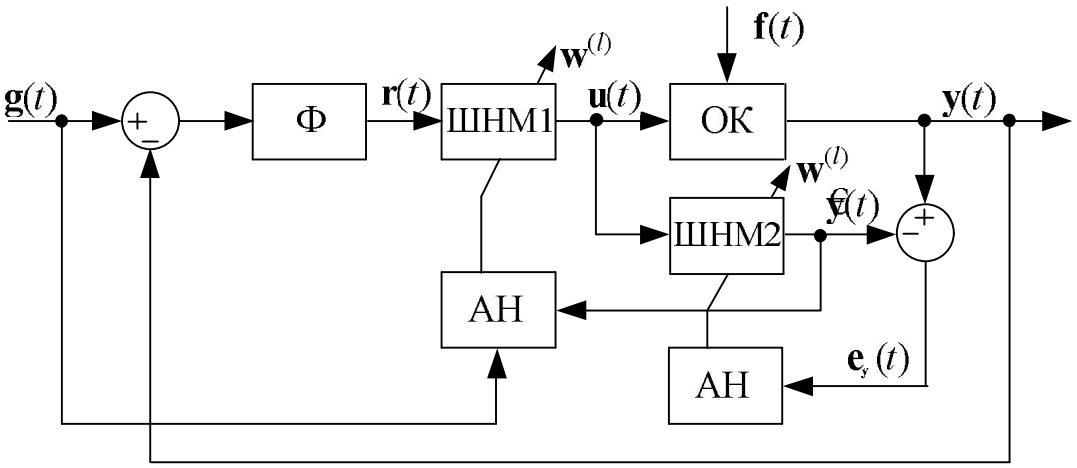


Рис. 1.8 – Схема адаптивної СУ з прямою та інверсною моделями об'єкта

В структуру такої адаптивної системи включено дві динамічні нейромережі, одна із яких (ШНМ1) виконує функції керуючого контролера, а ШНМ2 – НЕ. В АН ШНМ1 використовуються поточні виміри відтвореної функції $\mathbf{g}(t)$ і виходу прямої моделі ОК $\hat{\mathbf{y}}(t)$. Пряма модель виконується також у вигляді динамічної ШНМ2, навченої за помилкою (або прогнозом помилки) $\hat{\mathbf{e}}_1(t)$. Граничні динамічні властивості розглянутої системи випливають із наступних міркувань. У результаті навчання ШНМ2 помилка $\mathbf{e}_1(t) = \mathbf{y}(t) - \hat{\mathbf{y}}(t) \to 0$ при $t \geq 0$, так що $\hat{\mathbf{y}}(t) \to \mathbf{y}(t)$. При налагоджені ШНМ1 використається помилка навчання $\mathbf{e}_1(t) = \mathbf{g}(t) - \hat{\mathbf{y}}(t) \to 0$ при $t \to \infty$. Отже, $\mathbf{y}(t) \to \mathbf{g}(t)$ і $\hat{\mathbf{y}}(t) \to \mathbf{g}(t)$. Якщо A – оператор об'єкта, A_c – оператор адаптивного контролера ШНМ1 і $A_м$ – оператор моделі, реалізованої ШНМ2, то ланцюжок граничних переходів при навчанні мереж: $\hat{\mathbf{y}}(t) \to \mathbf{y}(t) \to \mathbf{g}(t)$ при $t \to \infty$ здійснюється тоді, коли $A_c = A_m^{-1} = A^{-1}$. В результаті налагодження ШНМ1 відображає інверсну модель об'єкта керування. Адаптивність системи обумов-

лена мінімумом апріорної інформації про об'єкт необхідної для структурного синтезу моделі, і можливістю підналагодження моделі й контролера (ШНМ2 і ШНМ1, відповідно) при неконтрольованих змінах динаміки ОК.

Існують також схеми розімкнутих адаптивних нейромережевих систем з нейроконтролером НК та налагоджуваною моделлю НЕ ОК, в цьому випадку НК використовується для формування необхідного сигналу керування при заданому вихідному стані об'єкта (якщо НК описує зворотну динаміку ОК [34]), а НЕ – для оцінки значень недоступних безпосередньому виміру параметрів об'єкта [34]. Ці дані використовуються потім для навчання НК.

Системи керування з ШНМ класифікуються як адаптивні системи [8, 34]. Вони також розділяються на замкнуті й розімкнуті адаптивні СК. Під замкнутістю СК розуміється пряме використання вхідних і вихідних сигналів об'єкта для адаптації керування, без проміжної ідентифікації об'єкта. Особливістю адаптивних СК є необхідність переналагодження нейроконтролера в темпі з реальним часом, а навчання можливе і в режимі попереднього навчання.

Найбільше часто використовуються замкнуті НСК з еталонною моделлю, структурна схема такої системи показана на Рис. 1.9 [34].

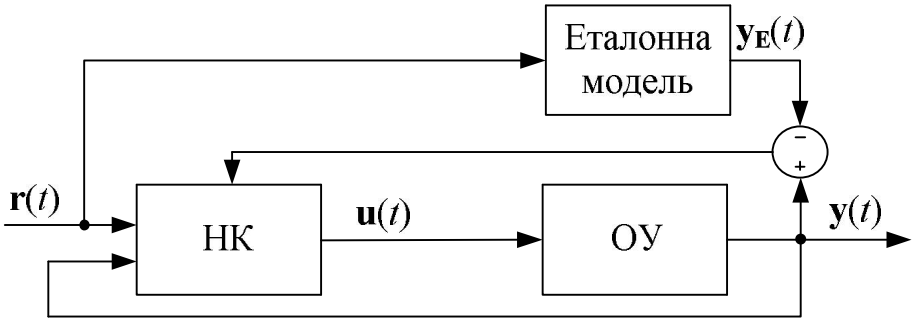


Рис. 1.9 – Схема НСК з еталонною моделлю

Метою навчання НК є одержання обмеженого керування $\mathbf{u}(t)$ такого, що

$$\lim \left\| y_E(t) - y(t) \right\| = 0 . \qquad (1.6)$$

У цьому випадку об'єкт буде відслідковувати бажану траєкторію, що визначена певною еталонною моделлю. Еталонна модель ОК може бути також

реалізована ШНМ навченою в режимі попереднього навчання. В [10] показано, що в такій схемі можливо побудувати адаптивний НК на базі ШНМ із контрольованим навчанням, що забезпечує СУ глобальну асимптотичну стійкість.

Наведені структурні схеми НСК, що навчаються у режимах попереднього навчання і навчання в реальному часі ілюструють основні відомі ідеї побудови СК з ШНМ [34]. Провівши огляд структурних схем СК з ШНМ можна зробити висновок що основними компонентами, таких систем, є пряма та інверсна модель ОК, компонент їх адаптації та компонент оцінки стану динамічної системи.

Не дивлячись на велику кількість переваг, НСК мають і велику кількість невирішених проблем, які стримують подальший їх розвиток і впровадження. Не розроблена загальна строга теорія по вибору типу і архітектури ШНМ для побудови моделей ОК, що приводить до необхідності пошукового підходу чи використання алгоритмів самоорганізації, що також потребує часових затрат. В роботі [35] описана загальна методика побудови нейромережевої моделі ОК. А також неможливість ввести апріорну інформацію в ШНМ іншим шляхом окрім навчання.

Основною проблемою нейронних мереж, при їх реалізації в СК, є великі затрати часу на їх навчання. Ця проблема обмежує можливості ШНМ при їх використанні в вбудованих СК технічними обєктами де є необхідність їх перенавчання в умовах реального часу в адаптивних та самоналагоджуваних СК. При реалізації нейромережевих компонентів на основі паралельних обчислювальних систем ця проблема може бути вирішена за рахунок:

- використання ШНМ мінімальної структури;

- використанням простих в обчислювальному сенсі алгоритмів навчання;

- вбудовування в структуру нейромережі елементів, що реалізують розпаралелену процедуру функціонування та навчання елементів нейромережі.

Система реального часу – це апаратно-програмний комплекс, на тривалість обробки інформації в якому накладаються обмеження зі сторони зовнішніх факторів.

Тривалість обробки інформації визначає реакцію системи на конкретну зовнішню подію, яка оцінюється проміжком часу від моменту отримання системою інформації до моменту закінчення його обслуговування. Ефективність та безпечність системи керування багато в чому визначається достатністю її швидкодії.

Швидкодією будемо називати період часу, протягом якого в системі виконується цикл (реалізується алгоритм) операцій: виявлення проблеми, її діагностика, вироблення цілей керування і керуючих впливів, прийняття рішень, передача і виконання рішень тощо.

При дослідженні СК має враховуватися можливість переходу об'єкта керування з одного стану в інший. Час реалізації алгоритму керування (t_y) не має бути більше, ніж час, протягом якого ОК переходить з поточного стану в неприпустимий. Тобто час розробки та реалізації керуючого впливу не має бути більше, ніж час, протягом якого об'єкт керування переходить з поточного стану в неприпустимий. В іншому випадку дії, що управляють будуть запізнюватися, ефективність керування – знижуватися, а ризики – зростати.

Вважається [36], що вихід будь-якого з параметрів $\mathbf{\Pi}_i\left(t+t_y\right)$ з області допустимих станів:

$$\mathbf{\Pi}_i^{\min}\left(t+t_y\right) < \mathbf{\Pi}_i\left(t+t_y\right) < \mathbf{\Pi}_i^{\max}\left(t+t_y\right)$$

переводить об'єкт керування в область неприпустимих станів:

$$\mathbf{\Pi}_i\left(t+t_y\right) < \mathbf{\Pi}_i^{\min}\left(t+t_y\right); \quad \mathbf{\Pi}_i\left(t+t_y\right) > \mathbf{\Pi}_i^{\max}\left(t+t_y\right).$$

Керування, що забезпечує знаходження об'єкта в області керованих станів з ймовірністю вище заданої (у тому числі при виникненні нештатних ситуацій), будемо називати керуванням в реальному масштабі часу.

Мета такого керування – забезпечити знаходження об'єкта керування в області допустимих або керованих станів. Система керування працює в реальному масштабі часу, якщо виконується співвідношення:

$$P_{ОД}\left(t,t_y\right) > P_{зад},$$

де: $P_{ОД}\left(t,t_y\right)$ – ймовірність знаходження об'єкта керування в області допустимих станів; $P_{зад}$ – задане мінімальне значення ймовірності знаходження об'єкта керування в області допустимих станів.

Імовірність знаходження об'єкта керування в області допустимих станів може бути визначена за допомогою формул:

$$P_{ОД}\left(t,t_y\right) = \min_i P_{\partial i}\left(t,t_y\right),$$

$$P_{Д}\left(t,t_y\right) = \int_{\mathbf{П}_{pi}^{\min}}^{\mathbf{П}_{pi}^{\max}} f\left(\mathbf{П}_{pi}\right)\cdot d\mathbf{П}_{pi},$$

де: i=1,…, n – номера параметрів об'єкта керування; t – поточний момент часу; $P_{\partial i}\left(t,t_y\right)$ – ймовірність знаходження об'єкта керування в області допустимих або керованих станів по параметру $\mathbf{П}_{pi}$; $\mathbf{П}_{pi}^{\max}$, $\mathbf{П}_{pi}^{\min}$ – максимальне і мінімальне значення швидкості зміни i-го параметра, при якому в момент виконання команди керування об'єкт керування буде знаходитися в області допустимих станів; $\mathbf{П}_{pi}$ – швидкість зміни i-того параметра об'єкта керування; $f\left(\mathbf{П}_{pi}\right)$ – щільність розподілу швидкостей зміни i-го параметра (включаючи позаштатні ситуації).

Система керування сягатиме прогнозної ефективності тільки при виконанні необхідної умови – забезпечення ймовірності знаходження об'єкта керування в керованому стані не нижче заданої. Якщо ця умова не виконується, то потрібно зменшити: час реалізації алгоритму керування; динамічні характеристики об'єкта керування.

Під нейромережевими елементами мінімальної структури в загальному випадку розуміється багатошарова ШНМ з одним або двома прихованими

шарами. ШНМ з одним прихованим шаром може апроксимувати функції з заданою точністю за відсутності обмежень на кількість нейронів в шарі. В роботах [34, 38] дано теоретичне обґрунтування емпіричному спостереженню, що мережі з двома прихованими шарами забезпечують велику точність при обмеженій кількості нейронів в шарах, в порівнянні з мережами з одним прихованим шаром.

Під простими в обчислювальному сенсі алгоритмами навчання будемо розуміти алгоритми, які не мають складних нелінійних функцій або коли процедура навчання застосовується до обмеженої частини нейромережі.

Під розпаралеленою процедурою навчання та функціонування будемо розуміти використання одночасного навчання та функціонування ряду незалежних елементів ШНМ. Такі можливості з'являються при апаратно-програмній реалізації на паралельних обчислювальних системах.

Паралельні обчислювальні системи – це системи які містять декілька (два і більше) обчислювальних вузлів (процесорів, обчислювачів, арифметичних пристроїв), зв'язаних між собою логічно і фізично для спільної обробки інформації.

Час вирішення паралельних задач визначається ефективністю розпаралелення процесів з ціллю максимальної загрузки ресурсів обчислювальної системи.

Таким чином, задачі, які ставить сучасний етап розвитку НСК не можуть бути вирішені без розробки нових методів та засобів реалізації компонентів НСК.

Для побудови НСК необхідно провести огляд та аналіз основних типів ШНМ та їх властивостей, для реалізації на їх базі компонентів нейромережевих систем керування, таких як пряма та інверсна моделі ОК, алгоритмів їх адаптації, для синтезу компонента адаптації, та проаналізувати сучасні програмні, апаратні і апаратно-програмні платформи реалізації ШНМ в СК.

1.2 Аналіз основних типів штучних нейронних мереж та їх властивостей для реалізації компонентів нейромережевих систем керування

На сьогоднішній день розроблено та досліджено декілька десятків типів ШНМ [39 – 48]. Основними типами ШНМ, що застосовуються в СК, є три типа мереж, це мережі прямого розповсюдження, динамічні мережі Хопфілда та RBF-мережі. Розглянемо їх детальніше.

1.2.1 Аналіз та особливості багатошарових нейронних мереж прямого розповсюдження

У повнозв'язних гомогенних БНМПР виходи базових елементів кожного шару сполучені зі всіма входами всіх базових елементів наступного шару, а функція активації $f(s)$ приймається однаковою для всіх базових елементів нейронної мережі [42]. Структура багатошарових нейронних мереж прямого розповсюдження (БНМПР) зображена на Рис. 1.10.

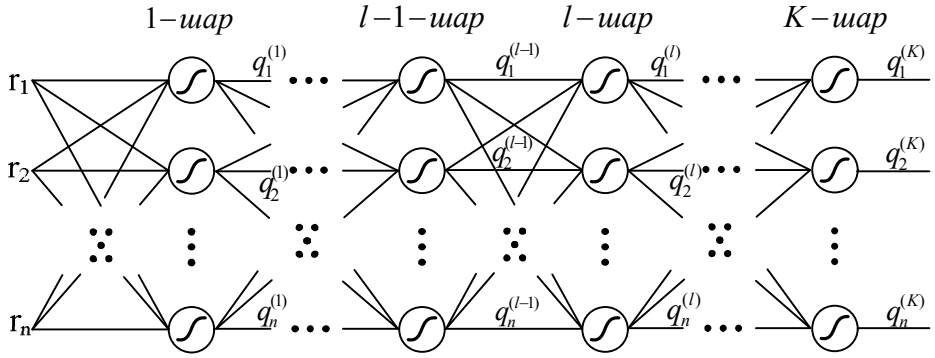


Рис. 1.10 – Структурна схема статичних БНМ

Проміжний l-й шар має n_i базових елементів. Зв'язки між базовими елементами в шарі відсутні. Виходи базових елементів l-го шару поступають на входи базових елементів тільки наступного $(l+1)$-го шару. Вихід i-го елементу в l-му шарі багатошарової нейромережі може бути визначений так само, як і для будь-якого базового елементу, у вигляді

$$q_i^{(l)} = f\left(\sum_{j=1}^{n_{l-1}} w_{i,j}^{(l)} q_1^{(l-1)} + w_{i,0}^{(l)} q_0^{(l-1)}\right) = f\left(s_i^{(l)}\right). \qquad (1.5)$$

У векторній формі вихід *l*-го шару мережі дорівнює

$$\mathbf{q}_1^{(l)} = \mathbf{f}\left(\mathbf{W}_1^{(l)}\mathbf{q}_1^{(l-1)} + \mathbf{w}_0^{(l)}\mathbf{q}_0^{(l-1)}\right), \tag{1.6}$$

де $\mathbf{w}_0^{(l)} = col(w_{1,0}^{(l)},...,w_{n_l,0}^{(l)})$ – вектор вагових коефіцієнтів $\mathbf{q}_0^{(l-1)}$ у шарі *l*, який для всіх $l = \overline{1,K}$ приймається рівним +1; $\mathbf{q}_1^{(l)} = col(q_1^{(l)},...,q_{n_l}^{(l)})$, $\mathbf{q}_1^{(l-1)} = col(q_1^{(l-1)},...,q_{n_{l-1}}^{(l-1)})$ – вектори виходів базових процесорних елементів шару *l* і виходів попереднього (*l*-1)-го шару, що поступають на входи базових елементів шару *l*. Для вихідного *K*-го шару маємо

$$\begin{aligned} \mathbf{q}_1^{(K)} &= \mathbf{W}_1^{(K)}\mathbf{q}_1^{(K-1)} + \mathbf{w}_0^{(K)}; \\ \mathbf{q}_1^{(K-1)} &= \mathbf{f}\left(\mathbf{W}_1^{(K-1)}\mathbf{q}_2^{(K-1)} + \mathbf{w}_0^{(K-1)}\right). \end{aligned} \tag{1.7}$$

Співвідношення (1.5) – (1.7) описують БНМПР з нелінійними функціями активації *f*(*s*) пошарово. Функцію $s_i^{(l)}$ на *i*-х виходах суматорів базових елементів шару *l* називають дискримінантною. У загальному випадку – це відрізок багатовимірного ряду Тейлора, його найбільший ступінь визначає порядок базових елементів. Так, співвідношення (1.5) відповідає виходу елементу першого порядку, а вектор вагових коефіцієнтів відповідного ряду Тейлора $\mathbf{w}_0^{(l)} = col(w_{i,0}^{(l)},...,w_{i,n_{l-1}}^{(l)})$ є «пам'ять» базового процесорного елементу. Дискримінантна функція є зважена сума входів БНМПР з коефіцієнтами, рівними значенням «вагів» $w_{i,j}^{(l)}$. Дискримінантна функція розділяє вхідний вектор мінімум на два класи у разі лінійної мережі з функцією першого порядку.

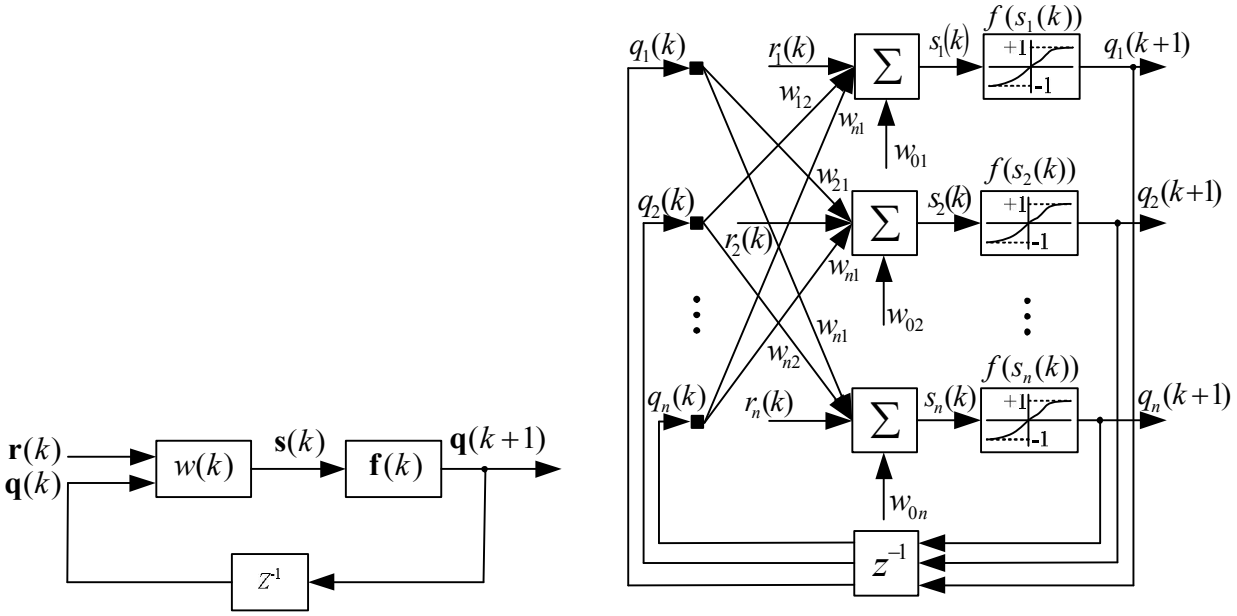
БНМПР завдяки своїм властивостям можуть бути використані для побудови регулюючих та коректуючих пристроїв, еталонної, адаптивної, номінальної та інверсно-динамічної моделей об'єкта, на основі яких виконується спостереження та оцінка параметрів об'єкта, спостереження та оцінка величини діючих в системі збурень, пошук або обчислення оптимальної програми зміни керуючого впливу, ідентифікація об'єкта, прогнозування стану об'єкта, розпізнавання, класифікації, аналіз великої кількісті даних, що надходять з високою швидкістю з великої кількості пристроїв та датчиків. При виконанні цих функцій в сучасній техніці основними параметрами є швидкість обробки

інформації, точність та кількість використаного ресурсу при апаратній реалізації. Тому підвищення ефективності роботи БНМПР при їх апаратній реалізації є актуальною задачею.

1.2.2 Аналіз та особливості динамічних багатошарових нейронних мереж

Динамічні багатошарові ШНМ відрізняються від БНМПР наявністю каналів, по яких інформація надходить і у зворотному напрямі, – від виходів на входи буферного шару [43 – 49]. Прийнято виділяти два класи багатошарових нейромереж зі зворотним розповсюдженням: рекурентні та рециркуляційні. У рециркуляційних розповсюдження сигналів між нейронами суміжних шарів здійснюється через двонаправлені зв'язки з різними ваговими коефіцієнтами в прямому і зворотному напрямах. Цей клас багатошарових мереж далі не розглядається, оскільки СК будуються на ланках однонаправленої дії. Рекурентні нейромережі також передають сигнали у зворотному напрямі, але по каналах зворотного зв'язку. На даний час розроблено і досліджено ряд модифікацій рекурентних мереж [39], але найбільше поширення набули релаксаційні мережі із зворотними зв'язками, названі мережами Хопфілда [40].

У мережах Хопфілда відбувається динамічне перетворення в часі вхідного сигналу $\mathbf{r} \in R^n$ в $\mathbf{q} \in R^{N_K}$. Стани нейронів мережі характеризують в кожен момент часу точку фазового простору і процес динамічного перетворення може бути інтерпретований як рух точки в цьому просторі до положення рівноваги. Стан мережі в положенні стійкої рівноваги характеризує інформацію, яку зберігає мережа. Якщо процес руху починається з неповної або спотвореної інформації $\tilde{\mathbf{r}} \in R^n$, то він йде по траєкторії, що веде до «найближчого» положення рівноваги. Це інтерпретується як операція асоціативного розпізнавання. На Рис. 1.11 показані структурні схеми рекурентної мережі Хопфілда: *а* – узагальнена; *б* – одношарова.



а) *б*)

Рис. 1.11 – Схеми рекурентної мережі Хопфілда: *а* – узагальнена; *б* – одношарова

У векторно-матричній формі статика мережі Хопфілда описується співвідношенням

$$\mathbf{s}(k) = \mathbf{W}\mathbf{q}(k) + \mathbf{r} - \mathbf{w}_0, \tag{1.8}$$

де компоненти вектора функції s(k) обчислюються як:

$$s_i(k) = \sum_{j=1, j\neq i}^{n} w_{i,j} q_j + r_i - w_{0i}, \quad i = \overline{1,n}.$$

Відповідно до цього матриця вагових коефіцієнтів W розмірністю $n \times n$ складається у вигляді симетричної матриці з нульовими елементами на головній діагоналі, тобто $w_{i,j}=w_{j,ii}\, w_{i,j} = 0$. Динаміка мережі Хопфілда забезпечується введенням зворотних зв'язків з виходів $q_j(k)$ на входи базових елементів ($i \neq j$) через елементи затримки $z^{-1} = e^{-\Delta ts}$ (Δt – період дискретизації безперервних функцій $q_j(k)$). В цьому випадку мережа Хопфілда відповідає нелінійній багатозв'язковій системі з векторними зворотними зв'язками, рисунок 1.11*а*. У векторній формі вихід дискретної моделі мережі Хопфілда записується у вигляді:

$$\mathbf{q}(k+1) = \mathbf{f}(\mathbf{q}(k), \mathbf{r}(k)), \quad k = 0,1,...; \tag{1.9}$$

$\mathbf{f}(.)=col(f(s_1), ..., f(s_n))$ – вектор нелінійних функцій активації. В (1.9) для k=0 $r_i(0)\neq 0$, а для $k>0$ всі $r_i(k)=0$. На виходах нейронів в результаті такої початкової імпульсної дії буде реакція $q_i(1)$, ($i=\overline{1,n}$). При r(k)= 0 (k = 1, 2,…) через дію від'ємного зворотного зв'язку з оператором затримки z^{-1} вектор виходу мережі Хопфілда буде:

$$\mathbf{q}(k+1) = \mathbf{f}(\mathbf{q}(k)). \qquad (1.10)$$

Співвідношення (1.9) та (1.10) – нелінійні різницеві рівняння 1-го порядку і з позицій теорії керування мережа Хопфілда може розглядатися як нелінійна динамічна ланка перетворення вхідного вектора $\mathbf{r}(k)$. Розрізняють дискретні і безперервні моделі мереж Хопфілда [39].

Складна нелінійна динаміка додає мережі Хопфілда можливості для формування на її виходах управляючих впливів, що є результатом взаємодії нелінійної функції перетворення $\mathbf{r}\in R^n$ в $\mathbf{q}\in R^m$ з алгоритмом налагодження вагових коефіцієнтів $w_{i,j}$ ($i,j=\overline{1,n}$). Присутність зворотних зв'язків породжує проблему стійкості мережі або, для дискретних моделей (1.9), (1.10), проблему збіжності послідовності $q_i(k) \xrightarrow{k\to\infty} q_i^* \in \{0,1\}$. Докладний аналіз проблеми стійкого навчання нейронної мережі Хопфілда розглянуто в [39].

Хоча більша частина літератури зосереджена на використанні мережі Хопфілда для вирішення знаменитої задачі Комівояжера, є ряд практичних проблем, які можна вирішити за рахунок таких ШНМ. Мережі Хопфілда мають два основних застосування. По перше, мережа використовується як асоціативна пам'ять, оскільки вона здатна запам'ятовувати, а потім відновлювати навіть при неповній вхідній інформації різні вектори $\tilde{\mathbf{r}}\in R^n$, що доречно при аналізі інформації з численних датчиків. Ця здатність забезпечується стійкістю досягнутого стану після налагодження мережі. В другому випадку мережа Хопфілда може бути використана для вирішення задач ідентифікації та керування.

Більшість застосувань мережі Хопфілда, для вирішення цих проблем, були отримані шляхом моделювання поведінки нейронної мережі Хопфілда

на звичайний комп'ютер [24, 25]. Однак, час обчислень надзвичайно повільний, хоча алгоритми генерують хороші рішення. При використанні ШНМ Хопфілда до практичних задач в вбудованих системах, ефективність їх роботи при апаратній реалізації необхідно підвищувати.

1.2.3 Аналіз та особливості нейронних мереж з радіально-базисними функціями активації

RBF-мережі є також універсальними апроксиматорами і при незначних обмеженнях можуть бути застосовані для апроксимації будь-якої безперервної функції [50 – 60]. RBF-мережі по своїй структурі відносяться до двошарових мереж, в яких використовується прихований шар з фіксованим нелінійним перетворенням вектора входу з постійними ваговими коефіцієнтами. Цей шар здійснює статичне відображення вхідних змінних $\mathbf{r} \in R^n$ у нові змінні $\mathbf{q}_1^{(l)} = col(q_1^{(l)},..., q_m^{(l)})$. Другий, лінійний вихідний, шар «зважує» ці змінні з вагами, що налагоджуються, $w_i = col(w_{i,1}, w_{i,2},..., w_{i,m})$, таким чином, що, наприклад, *i*-й скалярний вихід RBF-мережі записується у вигляді

$$q_i^{(2)} = q_i = F(\mathbf{r}) = \sum_{j=1}^{m} w_{i,j} f_i(\mathbf{r}) + w_0 \cdot 1, \quad i = 1,2,...,K.$$

Структура RBF-мережі з вектором входу $\mathbf{r} \in R^n$ і вектором виходу $\mathbf{q}^{(2)} = \mathbf{q} \in R^n$ зображена на Рис. 1.12.

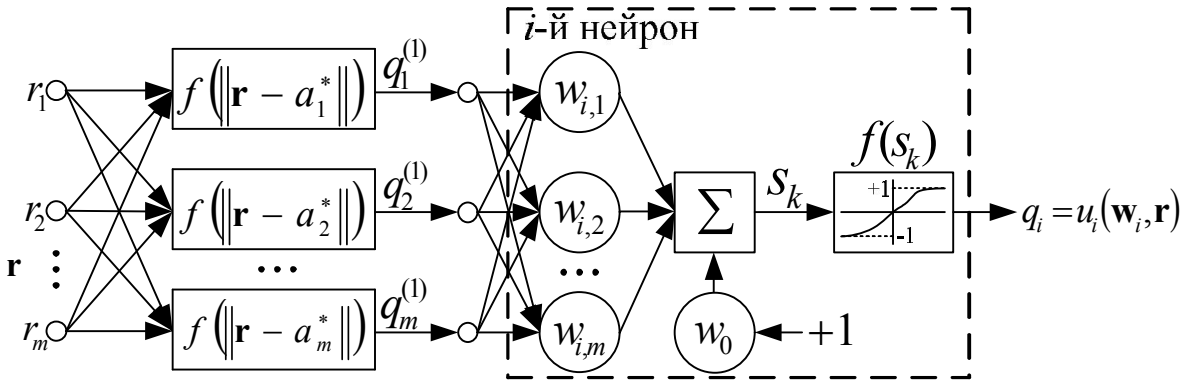


Рис. 1.12 – Схема RBF-мережі

Функції активації f_j(r):$R^{nn} \to R1$ вибираються в класі «радіальних» симетричних функцій:

$$f_i(\mathbf{r}) = f(\|\mathbf{r} - \mathbf{a}_j^*\|; h), \quad \mathbf{a}_j^* \in R^n.$$

Вони мають екстремум при зміні входів **r** тільки поблизу встановлених значень вагових коефіцієнтів БПЕ прихованого шару $\mathbf{w}_j^{(1)} = \mathbf{a}_j^*, j = 1,2,...,m;$, $\mathbf{w}_j^{(1)} = col(w_{j,1}^{(1)}, w_{j,2}^{(1)}, ..., w_{j,n}^{(1)})$.

Вагові коефіцієнти другого шару $w_{i,j} = w_{i,j}^{(2)}$, параметр h, положення так званих «центрів поля сприйнятливості» [50], що залежать від встановлених значень вагових коефіцієнтів в першому шарі $\mathbf{w}_j^{(1)} = \mathbf{a}_j^*$, форма функцій $f_j(\cdot)$ і їх число *m* в сукупності визначають властивості RBF-мережі.

В якості функцій *f*(*z*) вибирається одна з функцій:

$$\exp(-z^2);\ \exp(-z);\ 1/(1+z)^2;\ 1/1+z^2;\ z^\alpha \log z;\ z^2 + \alpha z + \beta.$$

На практиці найбільше використовується перша з наведених функцій – функція Гауса:

$$f_j(\mathbf{r}) = \exp\left(\sum_{k=1}^{n} 0.5\sigma_{j,k}^{-2} \left\|\mathbf{r} - \mathbf{a}_j^*\right\|^2\right),$$

де $a_j^* = col(a^*_{j,1}, a^*_{j,2}, \cdots, a^*_{j,n})$ – числовий вектор, а $\sigma_{j,k}$ – «ширина» функції Гауса.

Специфіка RBF-мережі полягає у тому, що в робочому режимі в них налагоджуються тільки вагові коефіцієнти лінійного вихідного шару. Помилка апроксимації обчислюється безпосередньо на виході мережі так само, як і в БНМ, але налагодження тільки одного, лінійного по параметрах налагодження шару знімає проблему пошуку глобального мінімуму функціонала навчання (як правило, квадратичного) і сприяє швидкій збіжності процесу навчання мережі.

Проблемою RBF-мереж є вибір числа радіально-базисних функцій, необхідних для апроксимації, і ця обставина стає критичним чинником у використовуванні RBF-мереж в задачах ідентифікації і керування, де необхідна

інформація для визначення розмірності RBF-мереж як правило, відсутня. В [56, 57] відмічено, що число необхідних радіально-базисних функцій росте експоненціально із зростанням числа вхідних змінних, тобто в нашому випадку із зростанням *n*. Звідси витікає висновок про застосовність RBF-мереж в тих задачах, де розмірність вхідної множини **r** обмежена і відома (в БНМ таке обмеження відсутнє).

У роботі [61] пропонується нейронна RBF-мережа з вихідними функціями гауса для навчання з безперервними вхідними сигналами. Вхідні сигнали вводяться в мережу на радіально-базисні функції, а потім виходи радіально-базисних функцій надходять на нейрони з сигмоїдальними функціями активації. Після навчання, на основі зворотного розповсюдження, локалізовані сигнали мережі RBF інтегруються і реконструюється у відповідний простір для даного навчання у шар нейромережі на основі сигмоїдальної функції. RBF-мережа такої архітектури має обидві переваги локальне навчання та здатність до глобального узагальнення.

Отже, ШНМ типу RBF можуть застосовуватися для вирішення різних різних задач, які можуть потребувати або не потребувати навчання в реальному часі. Серед цих задач можна виділити використання мереж RBF для класифікації інформації, розпізнавання зображень, формування функцій керування, а також інших застосунках. Для задач, які потребують високої швидкодії самої RBF-мережі а також її навчання, швидкість є основним визначальним фактором життєздатності спеціального обладнання. Такими задачами можуть бути вбудовані системи розпізнавання, класифікації, аналізу великої кількісті даних, що надходять з високою швидкістю з великої кількості пристроїв та датчиків та формування функцій керування, прийняття рішень**.** Тому підвищення ефективності роботи ШНМ типу RBF при їх апаратній реалізації є актуальною задачею.

1.3 Огляд та аналіз алгоритмів навчання багатошарових нейронних мереж

Навчання ШНМ є процес налагодження вагових коефіцієнтів $w_{i,j}$ її базових елементів, результатом чого є виконання мережею конкретних задач – розпізнавання, оптимізації, апроксимації, керування. Досягнення подібних цілей формалізується критерієм якості Q, мінімальне значення якого відповідає якнайкращому вирішенню поставленої задачі [62 – 66].

Різноманіття алгоритмів навчання визначається функціональним призначенням мережі, її архітектурою і вибраною стратегією навчання. Розрізняють три основні стратегії навчання: «з вчителем», з самонавчанням і змішану. У першому випадку ШНМ налагоджується по заданій навчальній вибірці відповідно до прийнятого правила або алгоритму. У другому випадку наперед не вимагається знати правильний результат навчання і в процесі налагодження вагових коефіцієнтів утворюється внутрішня структура активованих базових елементів, що відповідає пред'явленому вектору входу мережі. При змішаній стратегії навчання частина вагових коефіцієнтів $w_{i,j}$ налагоджується по заданій навчальній вибірці, а інша – відповідно до правил самонавчання.

Перерахованим стратегіям відповідають конкретні алгоритми навчання ШНМ. Кожен алгоритм орієнтований на конкретну архітектуру мережі і вирішення певного типу задач. Так, ШНМ для вирішення задач класифікації, апроксимації і керування навчаються по першій стратегії. Самонавчання використовується в мережах Хопфілда, мережах Кохонена, а змішана стратегія навчання застосовується в RBF-мережах.

Методи, використовувані при навчанні ШНМ, багато в чому аналогічні методам визначення екстремуму функції кількох змінних та діляться на 3 категорії – методи нульового, першого і другого порядку.

У методах нульового порядку для знаходження екстремуму використовується тільки інформація про значення функції в заданих точках.

У методах першого порядку використовується градієнт функціоналу помилки по налаштованим параметрами

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \mathbf{g}_k$$

де $\mathbf{x}_k$ – вектор параметрів; α_k – параметр швидкості навчання; $\mathbf{g}_k$ – градієнт функціоналу, відповідні ітерації з номером k.

Вектор в напрямку, протилежному градієнту, вказує напрямок найкоротшого спуску по поверхні функціоналу помилки. Якщо реалізується рух у цьому напрямку, то помилка буде зменшуватися. Послідовність таких кроків призведе до значень параметрів ШНМ, що забезпечують мінімум функціоналу. Труднощі викликає вибір параметра швидкості навчання α_k. При великому значенні α_k збіжність буде швидкою, але існує небезпека пропустити рішення або піти в неправильному напрямку. При малому кроці, ймовірно, буде обрано вірний напрям, але буде потрібно багато ітерацій. Параметр швидкості навчання може бути постійним або змінним. Правильний вибір α_k залежить від конкретного завдання і зазвичай здійснюється досвідченим шляхом; в разі змінного параметра його значення зменшується у міру наближення до мінімуму функціоналу.

В алгоритмах сполученого градієнта [64] пошук мінімуму виконується вздовж сполучених напрямків, що забезпечує зазвичай більш швидку збіжність, ніж при найшвидшому спуску. Ваги алгоритму сполучених градієнтів на першій ітерації починають рух у напрямку антиградієнта

$$\mathbf{p}_0 = -\mathbf{g}_0 .$$

Потім напрямок наступного руху визначається так, щоб він був пов'язаний з попереднім. Відповідний вираз для нового напрямку руху є комбінацією нового напрямку найшвидшого спуску і попереднього напрямку:

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k \mathbf{p}_{k-1}$$

Тут $\mathbf{p}_k$ – напрямок руху, $\mathbf{g}_k$ – градієнт функціоналу помилки, β_k – коефіцієнт відповідний ітерації з номером k. Коли напрямок спуску визначено,

то нове значення вектора параметрів, що настроюються $\mathbf{x}_{k+1}$ обчислюється за формулою

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

Методи другого порядку вимагають знання других похідних функціоналу помилки. До методів другого порядку належить метод Ньютона. Основний крок методу Ньютона визначається за формулою

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \mathbf{H}_k^{-1}\mathbf{g}_k$$

де $\mathbf{x}_k$, – вектор значень параметрів на k-й ітерації; $\mathbf{H}$ – матриця других приватних похідних цільової функції, або матриця Гессе – вектор градієнта на k-й ітерації. У багатьох випадках метод Ньютона сходиться швидке, ніж методи спряженого градієнта, але вимагає великих витрат через обчислення матриці Гессе. Для того щоб уникнути обчислення матриці Гессе, пропонуються різні способи її заміни наближеними виразами, що породжує так звані квазіньютонові алгоритми (алгоритм методу січних площин OSS [65], алгоритм LM Левенберга-Марквардта [66]).

В багатьох випадках застосування градієнтних методів локального пошуку, традиційно використовуваних для побудови нейромоделей, у ряді випадків, таких як багатоекстремальність цільової функції, недиференційованість функцій активації нейроелементів та інші є неприйнятним або неможливим [67 – 71]. Тому доцільним є використання методів еволюційного пошуку [71 – 83], які не вимагають обчислення значень похідних цільових функцій, що дозволяє ефективно застосовувати їх для вирішення задач побудови нейромоделей. Методи еволюційної оптимізації, на відміну від градієнтних методів, є методами глобального пошуку, у той час, як градієнтні методи, як правило, знаходять локальний оптимум, розташований в околі початкової точки пошуку.

Методи еволюційної оптимізації використовуються для вирішення різних завдань, пов'язаних із синтезом нейромереж: відбір ознак, настроювання ваг, вибір оптимальної архітектури мережі, адаптація навчального правила,

ініціалізація значень вагових коефіцієнтів, пошук оптимальних значень параметрів досліджуваної системи за наявною нейромоделлю і т.п. [70 – 76]. Незважаючи на те, що еволюційна оптимізація може виконуватися довше в порівнянні із градієнтними методами, вона, у загальному випадку, є значно менш чутливою до початкових параметрів навчання.

До еволюційних методів відносять: генетичні методи, еволюційні стратегії, генетичне програмування та еволюційне програмування. Кожний з таких методів має велику кількість різновидів. Для вирішення оптимізаційних задач, що виникають у процесі синтезу нейромережевих моделей [31, 34, 38], доцільним є використання генетичних методів, які є методами еволюційного пошуку й поєднують комп'ютерні методи моделювання генетичних процесів у природних і штучних системах.

Для застосування еволюційного пошуку до параметричного синтезу нейромереж необхідно: визначити цільову функцію й вибрати спосіб подання значень ваг у хромосомі. При виборі фітнес-функції дія параметричного синтезу нейромоделей враховують два фактори, такі як помилка між реальним і модельним виходом мережі та складність ШНМ. Цільова функція при параметричному синтезі нейромоделей використовується функція помилки, що обчислюється як середня різниця між реальним і модельним виходом мережі [80 – 82]. Хромосома при параметричному синтезі складається з K генів, що містять значення ваг і зсувів всіх нейронів мережі [83], схематично загальний вигляд хромосоми представлений на Рис. 1.12. При цьому для подання значень вагових коефіцієнтів у хромосомах застосовується дійсне кодування.

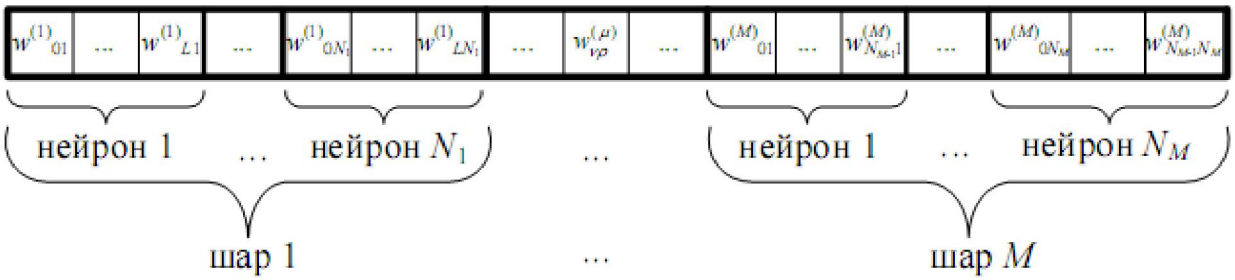


Рис. 1.12 – Схематичне подання хромосоми при параметричному синтезі нейромоделей

Розмір хромосоми визначається за формулою:

$$K = N_1\left(L+1\right) + \sum_{\mu=1}^{M} N_\mu\left(N_{\mu-1}+1\right)$$

де N_μ – кількість нейронів на μ-ому шарі; L – кількість ознак у навчальній вибірці; M – кількість шарів нейромережі.

Випадкове створення хромосом початкової популяції при параметричному синтезі нейромоделей приводить до зменшення ефективності процесу еволюційної оптимізації при пошуку оптимальних значень матриці ваг і збільшенню часу, необхідного для пошуку. Більш ефективним є рівномірній розподіл хромосом по всій області допустимих значень.

Еволюційна оптимізація синаптичних ваг нейромереж може бути виконана в послідовності описаній в [83].

Таким чином, навчання нейромоделей, засноване на еволюційному підході, не має потреби в обчисленні градієнта цільової функції й дозволяє знайти значення глобальних оптимумів синаптичних ваг для багатовимірних, полімодальних і недиференційованих цільових функцій.

Для вирішення задачі керування динамічними об'єктами за допомогою нейромережевих регуляторів та моделей потрібно щоб нейромережа навчалася в реальному часі, тобто потрібні алгоритми, які за найменший відрізок часу навчать мережу та можуть бути легко розпаралелені.

Однією з основних переваг застосування генетичного алгоритму в НСК динамічними об'єктами є можливість обрахунку функції помилки, що обчислюється як середня різниця між реальним і модельним виходом ОК, а не ШНМ.

Ще однією перевагою генетичного алгоритму при навчанні ШНМ є можливість застосування одного методу навчання для різних типів ШНМ.

Також генетичні алгоритми не потребують складних обчислень нелінійних математичних функцій, всі операції проходять з закодованими даними в хромосому – бітова послідовність. Операції з бітами добре реалізуються апаратними засобами.

Отже, зважаючи на всі переваги та недоліки існуючих на сьогодні методів навчання нейронних межеж, для апаратної реалізації компоненту налагодження ШНМ найкраще підходить генетичний алгоритм.

1.4 Аналіз сучасних програмних, апаратних і апаратно-програмних платформ реалізації штучних нейронних мереж в системах керування

За останні 20 років розроблена велика кількість різноманітних програмних, апаратних і програмно-апаратних платформ реалізації ШНМ. Реалізації за допомогою програмного забезпечення мають велику гнучкість, але більшість з них були реалізовані за допомогою архітектури Фон Неймана, яка є послідовною, всупереч притаманному паралелізму ШНМ. А також програмна реалізація за допомогою графічних процесорів GPU, що дозволила підвищити ефективність програм з ШНМ в десятки разів в порівнянні з реалізацією виключно на архітектурі Фон Неймана. Хоча збільшення продуктивності систем з послідовною архітектурою Фон Неймана та реалізація на GPU дозволяють реалізувати великі ШНМ, велика кількість застосунків ШНМ вимагають високої швидкості, компактності, низької вартості та енергоспоживання. Такі можливості з’являються з розвитком цифрових технологій, таких як програмовані логічні інтегральні схеми (FPGA), цифрові сигнальні процесори (DSP) та інтегральні схеми спеціального призначення (ASIC) [84–96]. Крім того, створення мов опису обладнання (HDL) дозволило інженерам розробити та кодувати модель ШНМ один раз та використовувати її на будь-якій апаратній платформі. ШНМ можуть бути реалізовані на будь-якому з цих пристроїв, що використовують VHDL [97 – 101]. Після написання коду VHDL він може бути використаний або для реалізації схеми в налаштованих пристроях, або може бути поданий на виготовлення чіпа ASIC. VHDL заснований на принципах програмного проєктування, але всупереч звичайним комп'ютерним програмам, які є послідовними, його опис за своєю суттю є паралельним.

Реалізовані програмно на комп'ютерах з послідовною архітектурою Фон Неймана нейросимулятори є дуже корисними інструментами для навчання та моделювання ШНМ. Застосування симуляторів, розроблених спеціально для вирішення задач різного роду за допомогою ШНМ, економить час, зусилля та підвищує якість моделювання при використанні стандартних алгоритмів. Крім того, графічний інтерфейс користувача існуючих нейросимуляторів роблять їх простими у використанні та допомагає користувачам моделювати процеси та системи з використанням ШНМ.

Одним з напрямнків програмної реалізації в останній час є реалізація за допомогою графічних процесорів GPU [102]. Завдяки параллельній архітектурі і спеціально вбудованим блокам реалізації ШНМ GPU підняли на новий рівень програмну реалізацію ШНМ. Використання GPU дозволило значно ускладнити реалізуємі системи на базі ШНМ, значно підвищити їх ефективність та скоротити в десятки разів швидкість навчання та обробки інформації такими системами в порівнянні з комп'ютерами з послідовною архітектурою Фон Неймана. А розробка спеціалізованих бібліотек реалізації программ з використанням ШНМ з підтримкою GPU дозволила значно спростити процесс розробки та дослідження програмних додатків з ШНМ. Однією з найпотужніших і простих у використанні бібліотек Python для розробки та оцінки програмних моделей глибокого навчання є Keras. Він обгортає ефективні чисельні бібліотеки обчислень Theano та TensorFlow [103].

На даний час для впровадження ШНМ використовувалось чимало апаратних технологій, але лише деякі досягли значного розвитку. FPGA, DSP та цифрові ASIC – ці технології, широко використовувані в апаратній реалізації ШНМ. Незважаючи на низьку гнучкість ASIC та їхню високу вартість виробництва, коли потрібно лише кілька мікросхем, ця технологія все ще залишається в перевазі при розробці апаратних за стосунків ШНМ через високий рівень ефективності з точки зору паралелізму, швидкості обробки інформації і низького споживання енергії. Ще одна тенденція впровадження апаратних засобів з ШНМ – використання пристроїв на базі DSP. DSP пропонують є

більш ефективний варіант через їх гнучкості та низької вартості, оскільки DSP можуть бути налаштовані лише завантаженням нової конфігурації в свою пам'ять. Однак FPGA, є найбільш ефективною цифровою платформою для впровадження ШНМ через їх низьку вартість на додаток до можливості впроваджувати схему в різних рівнях паралелізму.

Низька тактова частота DSP, в порівняні з FPGA, поки обмежує максимальну частоту оброблюваного аналогового сигналу до рівня в 10-20 МГц, але програмне керування DSP дозволяє досить легко змінювати не тільки режими обробки, але й функції, виконувані DSP. Крім обробки й фільтрації даних DSP можуть здійснювати маршрутизацію цифрових потоків, формування управляючих сигналів, сигналів системних шин ISA, PCI, тощо.

Перевагою FPGA-чіпів для реалізації нейропристроїв є висока швидкодія і постійне її зростання. Збільшення потужності FPGA дозволяє використовувати їх не тільки для реалізації простих контролерів і інтерфейсних вузлів, але й для цифрової обробки сигналів, складних інтелектуальних контролерів і нейрочіпів.

У FPGA-чіпів є можливість перепрограмування в системі. Програмування FPGA задає не тільки алгоритм обробки даних, але й сам тип пристрою, реалізованого чіпом, тому використання цієї функції може мати дуже великий ефект для гнучкості системи в цілому. З появою FPGA проєктування цифрових мікросхем перестало бути долею виключно великих підприємств з обсягами випуску в десятки і сотні тисяч кристалів. Проєктування і випуск невеликої партії унікальних цифрових пристроїв став можливий в умовах проєктно-конструкторських підрозділів промислових підприємств, в дослідницьких і навчальних лабораторіях. Промислово випускаються «заготовки» програмованих мікросхем з електричним програмуванням і автоматизованим процесом перекладу схеми користувача в послідовність імпульсів програмування роблять проєктування нових цифрових пристроїв порівнянним з розробкою програмного забезпечення. FPGA це дешева, проста та гнучка платформа, для впровадження та розробки нових цифрових апаратних засобів.

Вони також мають ряд специфічних переваг для реалізації ШНМ, наприклад, перепрограмовані FPGA дозволяють проводити прототипування, а також можна використовувати для вбудованих додатків, де надійність та простота нейронних обчислень найбільш потрібні навіть для низькомасштабного виробництва.

FPGA-системи характеризуються простотою апаратної реалізації нейроалгоритмів та нейромереж, наявністю програмних засобів високого рівня для побудови систем, сумісністю на рівні мови опису схеми, можливістю модифікації архітектури. Недоліками FPGA є з однієї сторони неповне використання ресурсів чіпа, з іншої – обмеженість ресурсів чіпа.

На основі цього аналізу можна зробити висновок, що нейромережеві системи можуть бути реалізовані кожним із перерахованих вище засобів, на Рис. 1.13 зображена порівняльна характеристика програмних та апаратних засобів реалізації ШНМ. Виходячи з того, що засоби реалізації нейромережевих систем повинні орієнтуватися на широке застосування в промислових умовах, бути універсальними і гнучкими, адаптуватися в темпі з реальним часом, бути простими і дешевими, тому найбільш перспективними засобами можна вважати FPGA для реалізації в системах керування та вбудованих системах. Далі розглянемо відомі результати реалізації ШНМ на FPGA.

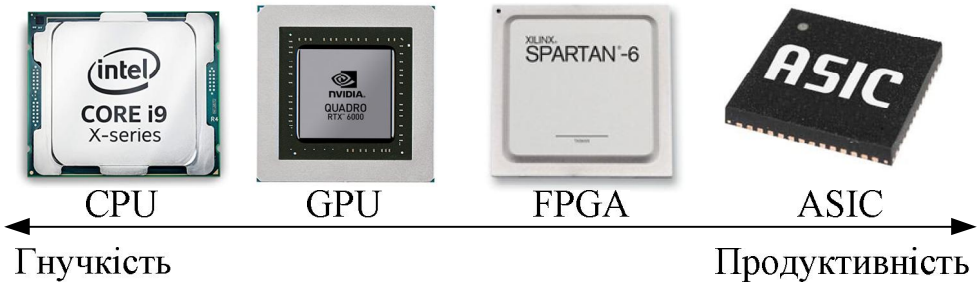


Рис. 1.13 – Порівняльна характеристика програмних та апаратних засобів реалізації ШНМ

В роботі [104] представлена ефективна апаратна реалізація вбудованого нейронного контролера на FPGA. Архітектура системи розроблена та описана за допомогою мови опису апаратних інтегральних схем. Для розробки

нейронного контролера використана мова опису VHDL та реалізований він на чіпі компанії Xilinx. При розробці нейронного контролера розглядаються питання, пов'язані з реалізацією нейрону з декількома входами та нелінійною функцією активації. В даній роботі розглянуті проблеми, що полягають у паралельній/послідовній реалізації нейронів, реалізації та точності даних для входів, ваг та функції активації, а також в використанні гібридного методу для наближення нелінійної, безперервної, монотонно зростаючої, диференційованої функції активації. Аналіз, моделювання, синтез, верифікація, прототипування та реалізація ШНМ контролера для систем керування реалізована на чіпі FPGA - SPARTAN6.

В роботі [105] представлена реалізація RBF-мережі, яка використовується для розпізнавання зображень. Представлено кілька реалізацій мережі на Xilinx Virtex 4 із показниками оцінки швидкості та займаного ресурсу чіпу FPGA.

В роботі [106] описана апаратна реалізація на FPGA RBF-нейромережі з використаннями чисел з фіксованою точкою. Представлено швидкодію RBF-нейромережі при апаратній реалізації на FPGA та використаний ресурс чіпу для різних форматів чисел з фіксованою точкою. Апаратна реалізація RBF-нейромережі виконана з використанням чіпу Virtex-6 xc6vcx240t-1ff1156.

В роботі [107] описана система машинного бачення в реальному часі, яка дозволяє локалізувати обличчя у відеопослідовностях та перевірити їх особу. Система машинного бачення побудована на методах обробки зображень та RBF-мережах. В роботі описано три апаратні реалізації система машинного бачення в реальному часі на вбудованих системах, що базуються, відповідно, на FPGA, на комп'ютерах з нульовим набором команд (ZISC) та на цифрових сигнальних процесорах (DSP).

В роботі [108]було використано FPGA для реалізації нейро-нечіткої мережі для вирішення нелінійних задач управління. Для отримання високошвидкісної роботи та додатків у режимі реального часу для розробки контролера нейро-нечіткого контролера було використано дуже швидку мову опису апа-

ратного забезпечення інтегрованої схеми (VHDL) та реалізовано на FPGA Xilinx Virtex 4. Експериментальні результати продемонстрували, що запропонована апаратна реалізація нейро-нечіткої мережі підтвердила життєздатність в управлінні системами автомобіля.

Як зазначалось в попередніх підрозділах, генетичні алгоритми та ШНМ вже давно використовуються в найрізноманітніших областях для оптимізації та пошуку задовільних рішень в технічних системах. Зовсім недавно було розширено спектр застосувань та варіацій генетичних алгоритмів, таких як паралельні та розподілені програми, апаратне забезпечення впровадження, нові пропозиції для генетичних операторів та гібридні (програмне та апаратне забезпечення). Проведемо огляд реалізацій генетичних алгоритмів на FPGA.

У роботі [109] розроблено модульну реалізацію генетичного алгоритму для FPGA. Однак реалізація не використовує стратегію паралельної обробки і витрачає кілька циклів для кожного покоління. У кожному поколінні необхідно читати та писати покоління з/у пам'ять.

Реалізації, розглянуті в [110, 111], є застосункоми генетичного алгоритму в системах цифрової обробки сигналів та управління, вбудованих в FPGA. Робота [110] представила генетичний алгоритм в реальному часі для програми адаптивного фільтрування з усіма модулями, реалізованими в апаратно, такими як функція фітнесу, відбір, кросовер, мутація та функції генератора випадкових чисел. В роботі була досягнута швидкість 320 тисяч поколінь в секунду. В роботі [111] запропоновано генетичний алгоритм для динамічних систем із декількома несучими, що базуються на блоках фільтрів.

Декілька підходів високошвидкісного апаратного забезпечення загального призначення для прискорення генетичних алгоритмів запропоновано в [112 – 114]. Ці підходи покращують конфігурацію параметрів генетичних алгоритмів в апаратній архітектурі. Але зменшують розпаралелювання апаратної частини та зменшують швидкодію генетичного алгоритму.

Для подальшого впровадження вбудованих систем зі штучним інтелектом на базі ШНМ в різноманітні системи військової техніки, зокрема бойо-

вих літаків, кораблів, безпілотників та систем протиповітряної оборони, в системах автопілоту автомобіля та інших, потрібно збільшувати ефективність їх роботи. А саме зменшувати час обробки інформації такими системами, час їх навчання, точність їх роботи, швидкість навчання та використаний апаратний ресурс. Ці показники є основними визначальними факторами життєздатності спеціального обладнання зі штучним інтелектом. Тому задача розробки нових методів, алгоритмів та засобів апаратної реалізації буде розглянута в цій роботі.

1.5 Висновки до 1 розділу

Проведено огляд та аналіз існуючих методів та засобів побудови НСК динамічними об'єктами та галузі їх застосування, аналіз основних типів нейронних мереж для подальшої реалізації на їх основі компонентів НСК, аналіз методів навчання нейронних мереж, аналіз сучасного стану програмних, апаратних і апаратно-програмних платформ реалізації ШНМ, в результаті якого сформульовані вимоги до обчислювальних систем, орієнтованих на їх реалізацію.

Таким чином є актуальна науково-технічна задача, зв'язана з підвищенням ефективності функціонування та навчання нейромережевих компонентів систем керування динамічними об'єктами, що забезпечують адаптацію та самоналагодження систем керування в реальному часі, при їх реалізації на сучасній елементній базі – програмованих логічних інтегральних схемах (FPGA).

Для вирішення вказаної задачі необхідно:

1. Розробити метод та алгоритми апаратної реалізації нейронних мереж на FPGA, для подальшого синтезу на їх базі компонентів нейромережевих систем керування.

2. Розробити метод проєктування апаратних компонентів нейромережевих систем керування.

3. Розробити метод проєктування апаратних компонентів контуру адаптації нейромережевих систем керування.

4. Побудувати та дослідити основні типи нейромережевих систем керування динамічними об'єктами на основі розроблених компонентів.

5. Розробити діючий макет нейромережевого контролера та впровадити розроблені технології, методи, засоби та алгоритми в систему керування динамічним об'єктом.

РОЗДІЛ 2. МЕТОД ТА АЛГОРИТМИ ПРОЄКТУВАННЯ ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ НА ОСНОВІ FPGA

2.1 Метод проєктування штучних нейронних мереж на FPGA

Технологія розробки застосувань на програмованих логічних інтегральних схемах основана на описі алгоритму такою мовою, як VHDL та автоматичній трансляції цього опису в опис на рівні логічних таблиць, та інших функціональних компонентів FPGA. Але елементарні функції в FPGA, як правило, реалізуються як окремі проєкти чи замовлені віртуальні модулі, в яких вказується розрядність даних та іноді – внутрішня будова.

Методи та алгоритми апаратної реалізації компонентів НСК мають бути орієнтовані на реалізацію в FPGA та бути адаптовані до їх елементного базису. FPGA різних фірм – виробників мають в своєму складі логічні таблиці, тригери, елементи суматорів, блоки ОЗУ/ПЗУ. Всі інші пристрої реалізуються на їх основі. Для конкретизації методів, алгоритмів та критеріїв їх оптимізації була вибрана архітектура FPGA фірми Xilinx. Параметри критеріїв, такі як апаратні затрати та затрати часу, можуть бути перераховані для інших виробників.

Однією з задач при апаратній реалізації нейронних мереж є реалізація штучного нейрона і його нелінійних активаційних функцій засобами FPGA. Існуючі підходи до цифрової реалізації нелінійних функцій використовують різні методи наближення, такі як табличний метод, ряд Тейлора, шматково-лінійне наближення тощо [107, 108, 115 – 118]. Ряди Тейлора потребують багатьох множень, а тому не є оптимальними для впровадження в FPGA, оскільки блок множення займає велику кількість ресурсів[119], результати такої реалізації представлені в [108]. Тож найкращим методом реалізації функцій активації нелінійного типу є шматково-лінійне наближення.

Модель нейрона працює в такий спосіб. Вхідні сигнали $a_{k,i}$ надходять на блоки, що реалізують функцію синапсів, кожний з яких характеризується

своїм ваговим коефіцієнтом w_{ki} (синаптичною вагою). Зважені вхідні сигнали подаються на лінійний суматор, після чого результат суми надходить на блок активаційної функції $f(.)$ і після відповідної обробки подається на вихід у вигляді сигналу q_k. Звичайно активаційна функція обмежує вихідний сигнал нейрона в діапазоні [0,1] або [-1,1]. Модель нейрона також містить у собі початкове зрушення b_k, що додається до вхідного сигналу блоку активаційної функції. Функціональна схема моделі штучного нейрона безперервного типу показана на Рис. 2.1.

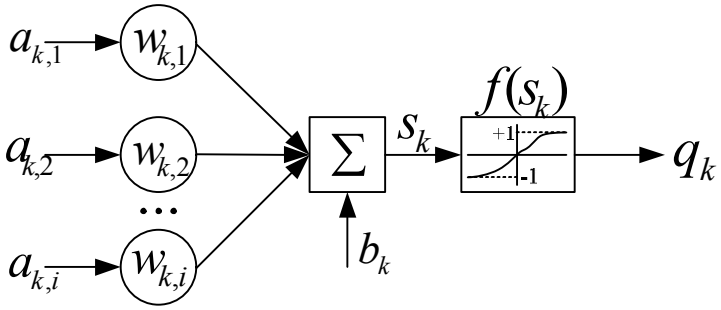


Рис. 2.1 – Схема штучного нейрона

Математично модель нейрона описується наступними залежностями:

$$q_k = f\left(S_k\right) = f\left(\sum_{i=1}^{n} w_{k,i}a_{k,i} + b_k\right),$$

де q_k – вихідний сигнал *k*-го нейрона; $f(.)$ – активаційна функція нейрона; $a_{k,i}$ – вхідні сигнали *k*–го нейрона; w_{ki} – синаптична вага *k*–го нейрона; b_k – зміщення *k*–го нейрона.

Активаційна функція $f(.)$ нейрона виконує нелінійне перетворення, здійснюване нейроном. Існує багато видів активаційних функцій, але найбільше поширені наступні чотири: гранична функція, лінійна функція, кусочно-лінійна функція, функції сигмоїдального типу, функція Гауса. Функції активації штучних нейронів представленні на Рис 2.2.

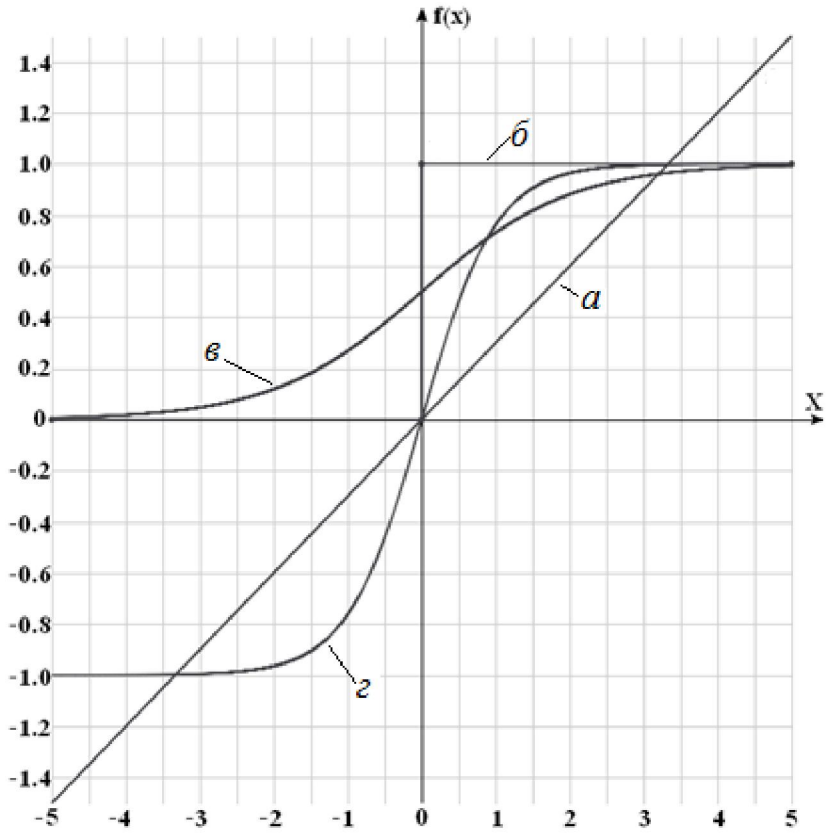


Рис. 2.2 – Функції активації штучних нейронів

Лінійна функція (Рис. 2.2 крива *а*) описується залежністю:

$$f(x) = px. \tag{2.1}$$

Порогова функція (Рис. 2.2 крива *б*) описується наступною залежністю:

$$f(x) = \begin{cases} 1, \text{якщо } x \geq a \\ -1, \text{якщо } x < -a \end{cases}, \tag{2.2}$$

де a – параметр, що визначає нахилу лінійної ділянки. При нескінченно великому значенні параметра a функція вироджується в порогову.

На Рис. 2.2 крива *б* зображений вид шматково-лінійної функції при a=1.

Сигмоїдальна функції (Рис. 2.2 крива *в*). Це найбільш ширше використовуваний тип активаційних функцій. Сигмоїдальні функції є монотонно зростаючими, безперервними і диференційованими. Диференційованість є важливою властивістю для деяких методів навчання і аналізу. Мають універ-

сальні апроксимаційні властивості [107, 115]. Особливістю нейронів з такою функцією активації є те, що вони посилюють сильні сигнали істотно менше, ніж слабкі, оскільки області сильних сигналів відповідають пологим ділянкам характеристики. Це дозволяє запобігти насичення від великих сигналів.

Під сигмоїдальними функціями розуміється клас функцій, які описуються виразом:

$$f(x,k,b,T,c) = k + \frac{c}{1+be^{Tk}}, \tag{2.3}$$

де x,k,b,T,c – параметри $k \in R; b \in R, b > 0; T,c \in R \setminus \{0\}$.

Якщо прийняти k=0, c=1, b=1, і T=-1, то вираз (2.3) прийме вигляд:

$$f(x,0,1,-1,1) = 0 + \frac{1}{1+1e^{-1x}} = \frac{1}{1+e^{-x}}. \tag{2.4}$$

Функція, описана виразом (2.4) називається «класичний» сигмоїд (рисунок 2.2 крива *в*).

Якщо прийняти k=1, c=-2, b=1, и T=2, то вираз (2.4) прийме вигляд:

$$f(x,1,1,2,-2) = 1 + \frac{-2}{1+1e^{2x}} = \frac{1+e^{2x}-2}{1+e^{2x}} = \frac{e^{2x}-1}{e^{2x}+1} = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}. \tag{2.5}$$

Функція, описана виразом (2.5) називається гіперболічний тангенс(Рис. 2.2 крива *г*).

Активною областю визначення функції активації нейрона є область значень вхідних параметрів, у якій спостерігається істотна зміна значень функції активації.

Для сигмоїдної функції використовується інтервал [–4; 4] як активна область визначення, при цьому функція приймає значення в інтервалі (0,018; 0,982), що становить 96,4 % від всієї області значень. Для тангенційної, сигмоїдної і радіально-базисної функцій як активну область визначення запропоновано в [34] інтервал [–2; 2], у якому зазначені функції приймають значення в інтервалах (–0,964; 0,964) і (0,0183; 1], відповідно.

Найскладнішою з погляду цифрової реалізації є сигмоїдальні функції активації, які є нелінійними функціями.

Запропонуємо наступний метод проєктування на FPGA функцій активації штучного нейрону. Вхідними даними для методу є функції що описуються формулою (2.9), або ті які можуть бути через них записані та точність з якою має бути реалізована функція активації.

На першому етапі виконується дослідження функції активації на симетричність відносно осей. Розглянемо функцію (2.4)

$$f(-x) = 1 - f(x) = 1 - \frac{1}{1+e^{-x}} = 1 - \frac{1}{1+\frac{1}{e^{x}}} = 1 - \frac{e^{x}}{e^{x}+1} = \frac{e^{x}+1}{e^{x}+1} - \frac{e^{x}}{e^{x}+1} = \frac{1}{e^{x}+1}$$

,

отримаємо:

$$1 - \frac{1}{1+e^{-x}} = \frac{1}{1+e^{x}}, \quad \text{або} \quad 1 - f(x) = f(-x). \tag{2.6}$$

Можемо розглядати $f(x)$ тільки для позитивних аргументів. Для його негативних значень можна знайти за формулою (2.6), що в свою чергу прискорить обчислення значення функції і зменшить займаний ресурс FPGA.

На другому етапі задаємо шматково-лінійну функцію на кожному з інтервалів $(-\infty; x_1), (x_1; x_2); \ldots (x_n; +\infty)$ окремої формулою. Записується це у вигляді:

$$f(x) = \begin{cases} k_0 x + b_0, x < x_1 \\ k_1 x + b_1, x_1 < x < x_2 \\ \cdots \\ k_n x + b_n, x_n < x \end{cases}, \tag{2.7}$$

На третьому етапі розрахунок значення *f*(*x*), для раніше розрахованого значення *x*, з пам'яті, вибираються коефіцієнти *k* і *b* формули (2.7), для негативних аргументів, значення функції розраховується за формулою (2.6).

Таким чином, на основі лінійних формул можемо знайти апроксимацію функції сигмоїдальної типу для будь-якого аргументу із заданою точністю.

Розроблено метод проєктування нелінійних функцій активації штучного нейрону на програмованих логічних інтегральних схемах, який відрізняється від [104] тим, що коефіцієнти шматково-лінійної апроксимації функції активації зберігаються у пам'яті тільки для позитивних або тільки для негативних значень аргументу, що дозволило оптимізувати кількість використаного обчислювального ресурсу та збільшити швидкодію обчислень нейронні мережі.

Для реалізації на FPGA запропонованого методу необхідно розробити алгоритми для різних функцій активації.

2.2 Розробка алгоритму апаратної реалізації штучного нейрона з сигмоїдальною функцією активації

Для апаратної реалізації ШНМ, для входів, ваг та функції активації необхідно враховувати формат чисел які будуть при цьому використані (плаваюча точка чи фіксована точка) та кількість бітів (точність), які можуть становити від 16 біт, 32 біт або 64 біт, формат чисел з плаваючою точкою або формат з фіксованою точкою. Підвищення точності, кількості біт на кожне число, значно збільшує використовувані ресурси.

Формат чисел з плаваючою точкою потребує значних апаратних ресурсів але має високу точність, тоді як формат чисел з фіксованою точкою потребує мінімальних ресурсів але має гірші показники точності. Мінімальна допустима точність та мінімально допустимий діапазон мають бути визначені для оптимізації використаного апаратного ресурсу та швидкості обчислень, не впливаючи на продуктивність ШНМ. Точність чисел з фіксованою точкою не залежить від числа, але залежить від положення точки, яка кількість чисел буде відведена на цілу частину та яка кількість на дробову. В роботах [104,105] показано, що потреба в апаратних ресурсах мереж, реалізованих за допомогою чисел з фіксованою точкою, помітно в два рази менша, ніж при використанні чисел з плаваючою точкою з аналогічною точністю та діа-

пазоном значень. Також обчислення за допомогою чисел з фіксованою точкою забезпечують кращу конвергенцію за меншими тактовими циклами. Для нормалізованого введення та виведення в діапазоні [0,1] для реалізації нелінійної функції активації 16-бітове представлення чисел з фіксованою точкою є допустимою точністю.

Запропонуємо алгоритм реалізації штучного нейрона з класичною сигмоїдальною функцією активації на FPGA з використанням мови VHDL.

Крок 1. Задаємо синаптичні ваги штучного нейрона.

Кожному нейрону задається блок оперативної пам'яті де зберігаються ваги. Для завдання ваг кожного нейрона використовуються такі сигнали:

–synaddr (synapse address) – вибір синапсу вага якого буде зчитуватися або записуватися за його номером у двійковому коді. Для нейрона на 4 синапси цей вхід буде 2-х бітним, на 8 - 3-х і т.д.;

– synsetw (synapse set weight) – значення ваги, яке треба записати в синапс;

– synwren (synapse write enabled) – сигнал при наявності 1 на вході якого нейрон записує в обраний синапс значення з synsetw, коли даний сигнал дорівнює 0, нічого не відбувається.

Дані змінні необхідні для подальшого навчання нейрону та нейронних мереж.

Крок 2. На входи штучного нейрона подаємо значення вхідного вектора, задаємо змінну *x*, рівної виходу суматора:

$$x = \sum_{i=1}^{N} w_i \cdot a_i,$$

де *a* – входи нейрона, *w* – синаптичні ваги нейрона.

Розрахунок проходить з використанням чисел з фіксованою точкою. Кожне число – 16 біт, з яких 9 приходить на цілу частину і 7 на дробову. Для арифметичних операцій над цими числами використовується бібліотека fixed_pkg_c.vhdl. Вона синтезується, арифметичні операції даної бібліотеки добре оптимізовані.

Крок 3. Розраховуємо модуль від аргументу сигмоїдальної функції, задаємо змінну x':

$$x' = |x|.$$

Крок 4. На даному кроці виконується розбиття на лінійні шматки сигмоїдальної функції активації, і визначаються коефіцієнти лінійних рівнянь k і b. Розбивка на лінійні шматки їх рівняння і похибка показана на Рис. 2.3.

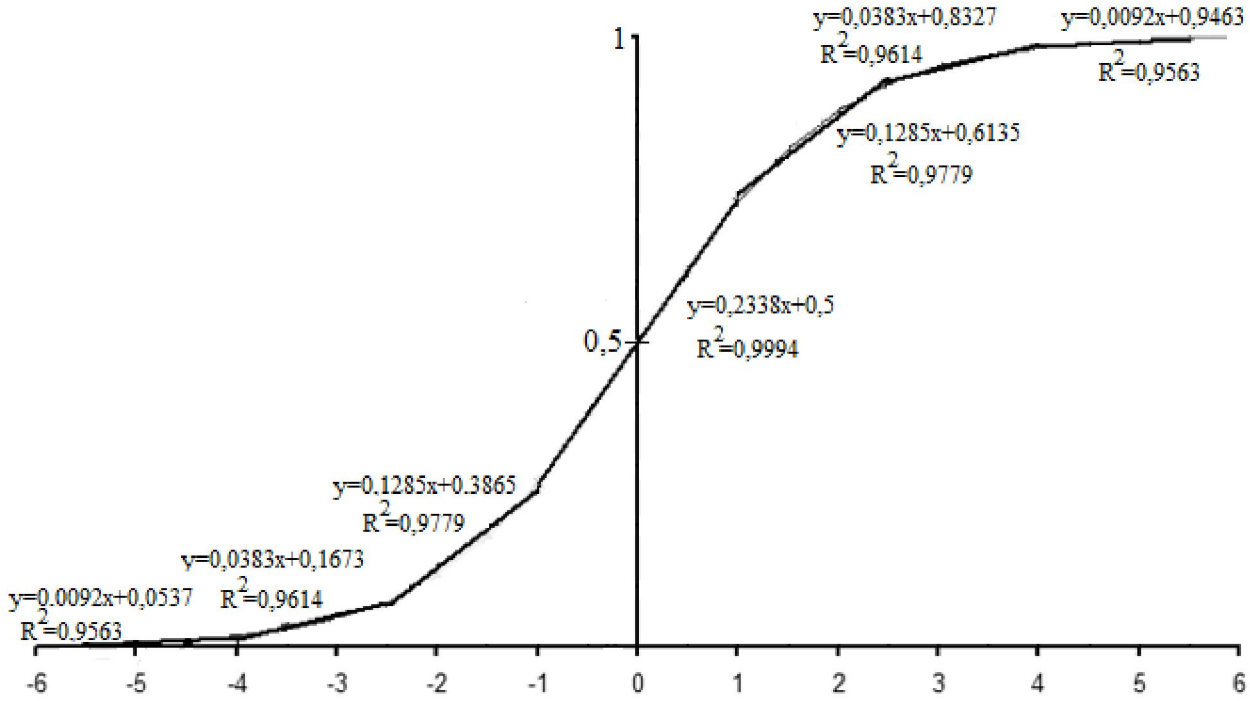


Рис. 2.3 – Шматково-лінійна апроксимація сигмоїдальної функції

Визначаємо коефіцієнти лінійних рівнянь:

$$k,b = \begin{cases} [0.234, 0.500], \textit{якщо } 0 \le x' < 1, \\ [0.129, 0.605], \textit{якщо } 1 \le x' < 2.5, \\ [0.234, 0.500], \textit{якщо } 2.5 \le x' < 4, \\ [0.009, 0.946], \textit{якщо } x' \ge 4. \end{cases}$$

Крок 5. Визначаємо змінну f, обчислюємо значення за формулою:

$$f = k \cdot x' + b.$$

Крок 6. Якщо $x < 0$, обрахувати значення локальної змінної за формулою:

$$f = 1 - f.$$

Крок 7. Задати значення вихідного сигналу нейрона рівним значенню змінної *f*.

На Рис. 2.4 представлено класичну сигмоїдальну функцію за формулою (2.4) та функцію реалізовану в FPGA за розробленим алгоритмом. Як видно реалізований обчислювальний блок сигмоїдальної функції в FPGA досить точно відображає її, для подальшої реалізації штучних нейронних мереж та компонентів нейромережевих систем керування на їх базі.

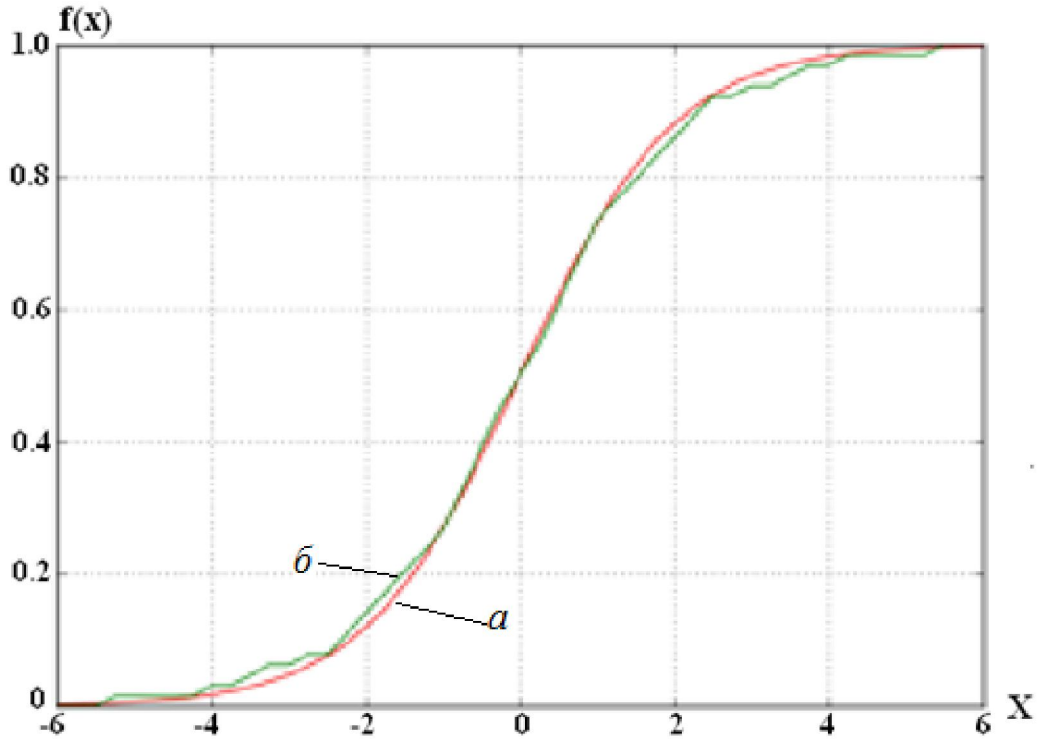


Рис. 2.4 – Графік сигмоїдальної функції, реалізованої в FPGA

В таблиці 2.1 представлені результати конфігурації сигмоїдальної функції на FPGA різних серій. В таблиці 2.1 серія та швидкодія (Speed) чіпу FPGA, швидкодія, як сумарна затримка комбінаційної схеми блоку обчислювача сигмоїдальної функції та використаний ним ресурс.

Таблиця 2.1 – Результати результати конфігурації сигмоїдальної функції на FPGA різних серій

FPGA		Швидкодія, нс	Ресурс FPGA (LUT)
Серія	Speed		
Xilinx	-5	26.073	75

Spartan 3	-4	30.202	
Xilinx Spartan 6	-2	20.451	60
	-3	17.501	
Virtex 5	-2	12.892	60
	-1	15.149	
Virtex 4	-12	15.3	75
Xilinx Zinq	-1	7.996	60
	-2	9.111	
	-3	10.892	
Virtex 7	-1	10.118	60
	-2	8.461	
	-3	8.461	
Virtex 6	-2	9.297	60
	-1	10.781	
Kintex 7	-1	7.429	60
	-2	8.461	
	-3	10.118	
Artix 7	-1	12.465	60
	-2	10.495	
	-3	9.124	

В порівнянні отриманих результатів з результатами з найблищого аналогу [104], ресурс FPGA (LUT) – 108, швидкодія 22.12нс, як затримка обчислювального блоку, та максимальне відхилення 0,5%, по реалізації сигмоїдальної функції активації на чіпі сімейства Xilinx Spartan 6, зменшено використаний ресурс FPGA до 60 LUT, збільшена швидкодія до 17.5нс, як затримка обчислювального блоку, та максимальне відхилення зменшено до 0,45%. Зменшити кількість використаного ресурсу вдалось за рахунок розробки нового методу шматково-лінійної апроксимації сигмоїдальної функції, що відрізняється використанням лише додатних значень аргументу функції, та перерахунку від'ємних по формулі (2), а також підвищити точність обрахунку функції, по 8 лінійним відрізкам. В порівняні з іншою роботою [120], ресурс FPGA (LUT) – 336, швидкодія 120.1 нс, як затримка обчислювального блоку, та максимальне відхилення 0,556%, по реалізації сигмоїдальної функції активації на чіпі сімейства Xilinx Spartan 3, зменшено використаний ресурс

FPGA до 75 LUT, збільшена швидкодія до 30.2нс, як затримка обчислювального блоку, та максимальне відхилення зменшено до 0,45%.

Реалізований за даним алгоритмом штучний нейрон з 4-ма входами і сигмоїдальною функцією активації на FPGA з використанням 16-розрядних чисел з фіксованою точкою зайняв 672 LUTs (Look Up Table – вентилі логічної матриці). Штучний нейрон розроблений на FPGA як окремий обчислювальній блок зображено на рисунку 2.5. Швидкодія, як сумарна затримка комбінаційної схеми блоку нейрона, склала 75.6 нс. Похибка абсолютна ± 0.005, точність реалізації сигмоїдальної функції показана на Рис. 2.5. При зменшенні або збільшенні кількості лінійних відрізків за формулою (5), кількості входів нейрона або зміну розрядності чисел буде змінений і використаний ресурс FPGA для одного нейрона, а також точність і швидкодія.

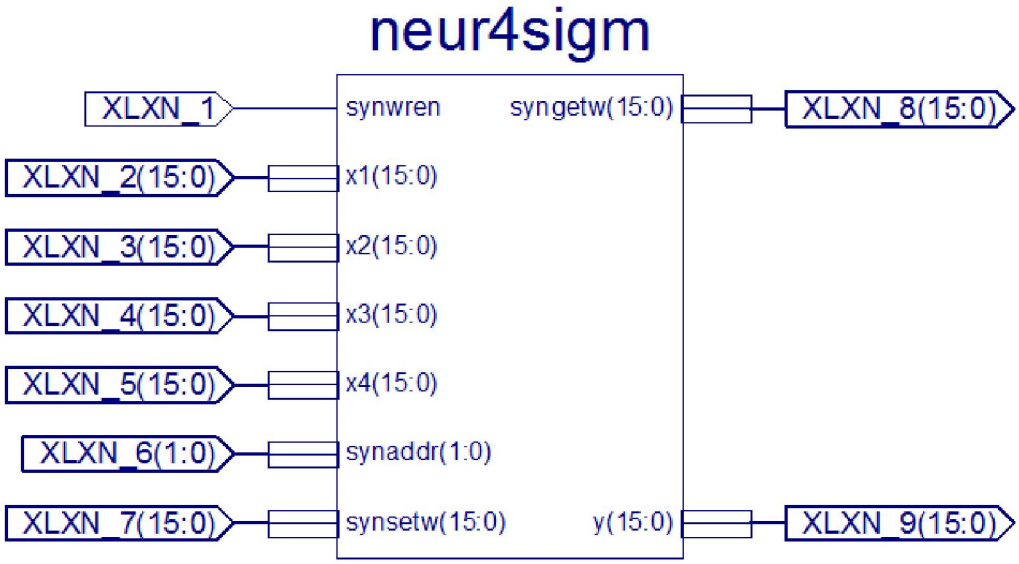


Рис. 2.5 – Блок штучного нейрона в FPGA

Проведено дослідження реалізації штучного нейрону за розробленим методом та алгоритмом у FPGA різних серій. В таблиці 2.2 показані результати такого дослідження.

Таблиця 2.2. – Результати моделювання штучного нейрону в FPGA різних серій

FPGA		Швидкодія, нс	Ресурс FPGA (LUT)
Серія	Speed		
Xilinx Spartan 3	-5	51.643	493
	-4	59.845	
Xilinx Spartan 6	-2	38.964	315
	-3	33.213	
	-3N	34.465	
Xilinx Zinq	-3	17.337	309
	-2	19.647	
	-1	23.493	
Virtex 7	-1	22.161	309
	-2L	18.529	
	-2	18.528	
	-2G	18.388	
Virtex 6	-2	20.288	309
	-1	23.477	
Virtex 5	-2	24.965	369
	-1	29.401	
Virtex 4	-10	37.714	485
	-11	32.903	
	-12	29.051	
Kintex 7	-3	16.360	309
	-2	18.528	
	-1	22.161	
Artix 7	-1	28.131	309
	-2	23.666	
	-3	20.629	

Результати моделювання сигмоїдальної функції та штучного нейрона на FPGA за розробленим методом та алгоритмом показали підвищення ефективності розроблених апаратних блоків в порівнянні з відомими аналогами.

2.3 Розробка алгоритму апаратної реалізації нейронів прихованого шару RBF-мережі

Як зазначалось вище RBF-мережі є універсальними апроксиматорами і при незначних обмеженнях можуть бути застосовані для апроксимації будь-якої безперервної функції. RBF-мережі по своїй структурі відносяться до двошарових мереж, в яких використовується прихований шар з фіксованим нелінійним перетворенням вектора входу з постійними ваговими коефіцієнтами. Цей шар здійснює статичне відображення вхідних змінних $\mathbf{r} \in R^n$ у нові змінні $\mathbf{q}_1^{(l)} = col(q_1^{(l)},..., q_m^{(l)})$.

При апаратній реалізації RBF-мережі однією з основних проблем є реалізація прихованого шару та нелінійних функцій активації $f_i(r)$, зокрема реалізація функції Гауса.

При апаратній реалізації RBF-мережі однією з основних проблем є реалізація прихованого шару та нелінійних функцій активації $f_i(r)$, зокрема реалізації функції Гауса.

Функція Гауса представлена на Рис. 2, і описується формулою:

$$f(x) = ae^{-\frac{(x-b)^2}{2\sigma^2}} \tag{2.8}$$

де a відповідає за висоту функції, b – за переміщення в осі x, а σ – за швидкість спадання функції при віддаленні вектора від центру.

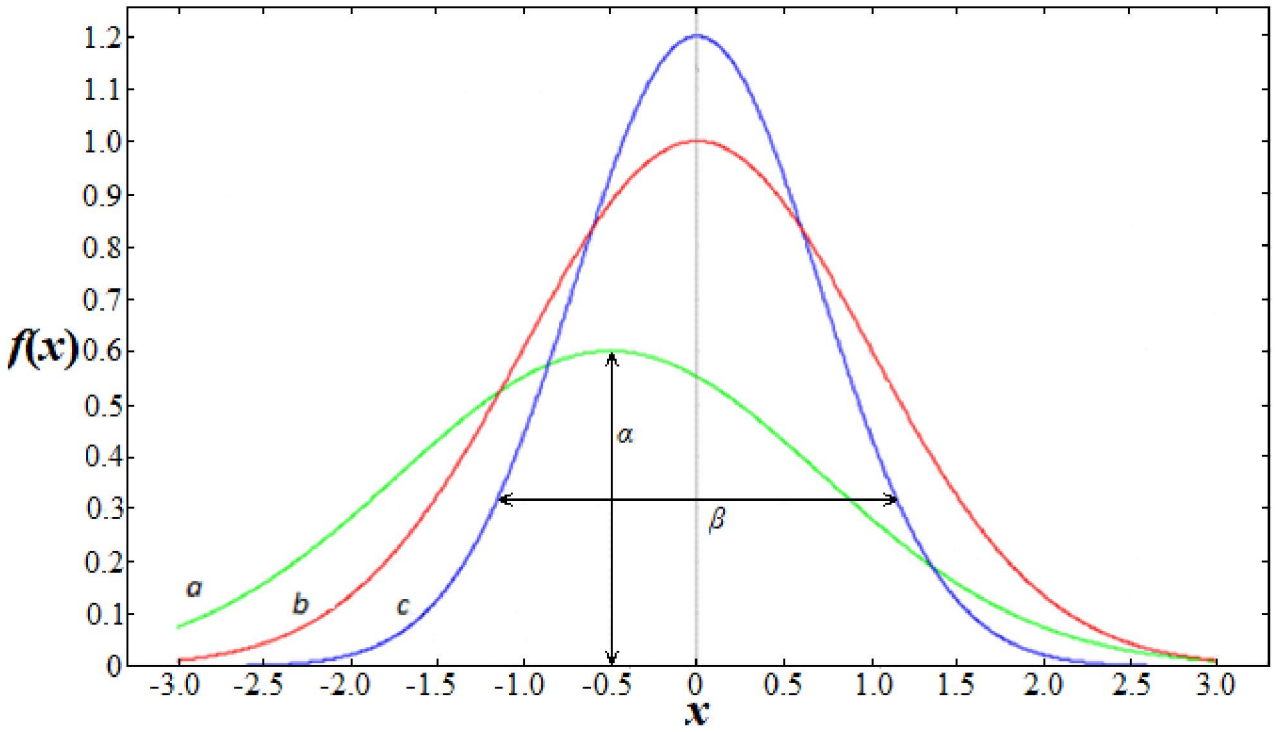


Рис. 2.6 – Графіки функції Гаусса

На Рис. 2.6 графік *a* функції Гауса має низьку швидкість спадання та зсув по осі OX і описується він наступним чином:

$$f(x) = 0.6e^{-\frac{(x+0.5)^2}{3}} \tag{2.9}$$

На Рис. 2.6 графік *c* функція більш видовжена і її математичний опис має такий вигляд:

$$f(x) = 1.2e^{-\frac{x^2}{1}} \tag{2.10}$$

Всі коефіцієнти функції підбираються при проєктуванні нейронної мережі під конкретну задачу в залежності від необхідного результату.

На Рис. 2.6 графік *b* функції Гауса описується він наступним чином:

$$f(x) = e^{-\frac{x^2}{2}} \tag{2.11}$$

Запропонуємо наступний покроковий алгоритм апаратної реалізації на FPGA штучного нейрона RBF-мережі з прихованим шаром з функціями Гауса на основі розробленого методу реалізації функцій активації. Алгоритм виконується наступним чином:

Крок 1. На входи штучного нейрона подаємо значення вхідного вектора, задаємо змінну *x*. Для реалізації використовуються числа з фіксованою комою. Числа 16-бітні, 9 біт приходить на цілу частину і 7 на дробову.

Крок 2. Розраховуємо модуль від аргументу функції Гауса, задаємо змінну x':

$$x' = |x| \tag{2.12}$$

Крок 3. На даному кроці виконується розбиття на лінійні шматки функції активації Гауса, і визначаються коефіцієнти лінійних рівнянь *k* і *b*. Розбивка на лінійні шматки їх рівняння показана на Рис. 2.10, графік *а* – функція Гауса, графік *б* – її шматково-лінійна апроксимація.

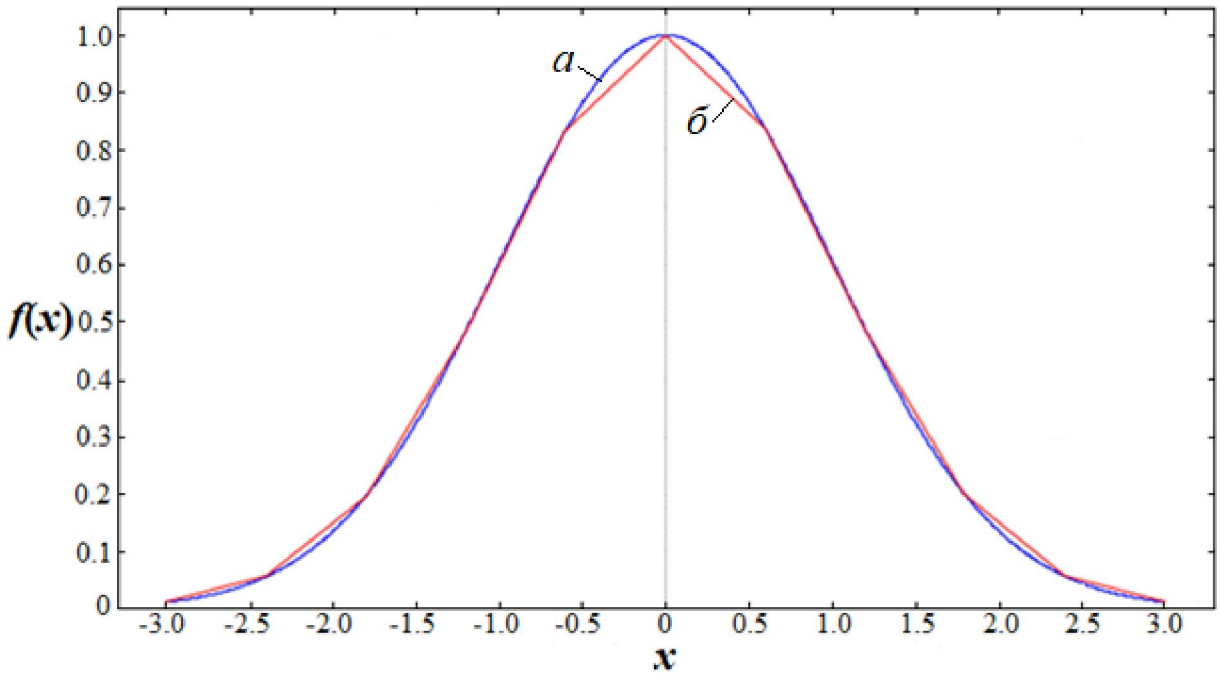


Рис. 2.10 – Шматково-лінійна апроксимація функції Гаусса

Визначаємо коефіцієнти лінійних рівнянь:

$$k,b = \begin{cases} [-0.275,\ 1], якщо\ 0 \le x' < 0.6, \\ [-0.58,\ 1.183], якщо 0.6 \le x' < 1.2, \\ [-0.48,\ 1.063], якщо 1.2 \le x' < 1.8, \\ [-0.238,\ 0.628], якщо 1.8 \le x' < 2.4, \\ [-0.093,\ 0.28], якщо\ x' \ge 2.4. \end{cases} \tag{2.13}$$

Крок 4. Визначаємо змінну *f*, обчислюємо значення за формулою:

$$f\left(x'\right) = k \cdot x' + b \tag{2.14}$$

Крок 5. Результат $f\left(x'\right)$ подається на нейрон з сигмоїдальною функцією активації.

Результати моделювання функції Гаусса в FPGA різних серій за розробленним способом представлені в таблиці 2.3. Похибка абсолютна складає ± 0.005 при реалізації функції Гаусса за допомогою 16-ти розрядних чисел з фіксованою точкою.

Таблиця 2.3 – Результати моделювання функції Гаусса в FPGA різних серій

FPGA		Швидкодія, нс	Ресурс FPGA (LUT)
Серія	Speed		
Xilinx Spartan 3	-5	27.053	106
	-4	29.51	
Xilinx Spartan 6	-2	19.35	40
	-3	18.31	
Virtex 5	-2	11.92	77
	-1	14.25	
Virtex 4	-12	16.44	104
Xilinx Zinq	-1	7.35	39
	-2	10.16	
	-3	10.59	
Virtex 7	-1	11.35	39
	-2	8.26	
	-3	8.73	
Virtex 6	-2	9.66	39
	-1	11.27	
Kintex 7	-1	7.89	39
	-2	8.66	
	-3	9.38	
Artix 7	-1	11.675	39
	-2	9.865	
	-3	8.78	

Загальний вигляд RBF-мережі з чотирма нейронами прихованого шару і одним нейроном з сигмоїдальною функцією активації зображений на Рис. 2.11.

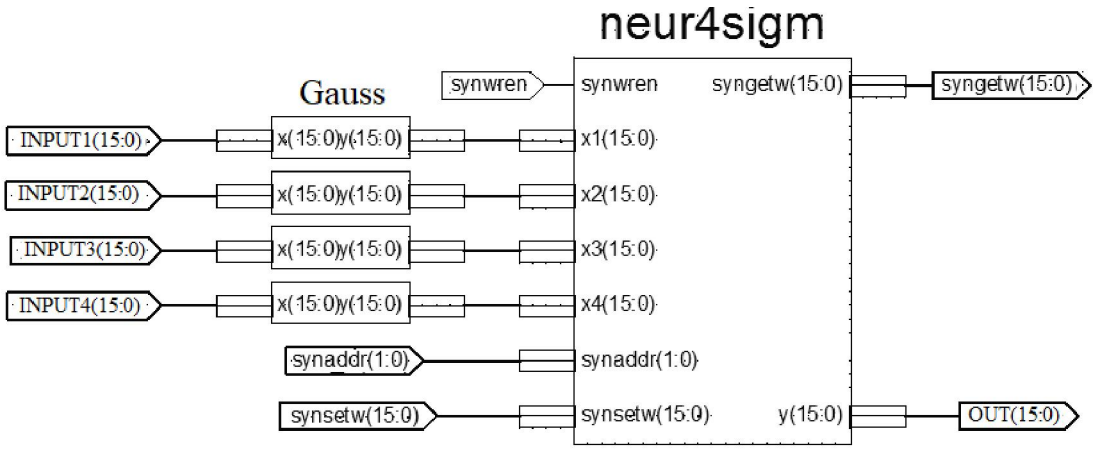


Рис. 2.11 – Схема нейронної мережі RBF реалізованої на FPGA

Реалізована за даним алгоритмом нейронна мережа з 4-ма нейронами прихованого шару та з одним нейроном з сигмоїдальною функцією активації на FPGA з використанням 16-розрядних чисел з фіксованою точкою зайняла 1089 LUTs. Кожний нейрон прихованого шару RBF-мережі розроблений на FPGA як окремий обчислювальній блок. Швидкодія, як сумарна затримка комбінаційної схеми блоку RBF-мережі склала 117.805 нс. З цих результатів видно, що кількість LUTs зросла на 63% при порівнянні з одним нейроном сигмоїдальної функції активації, тоді як час збільшився лише на 50%, це пов’язано з тим, що нейрони прихованого шару можуть обчислюватись паралельно. Синтез та моделювання нейронної мережі з 4-ма нейронами прихованого шару та з одним нейроном з сигмоїдальною функцією активації виконано на програмному забезпеченні Xilinx ISE Design Suite 13.2 та ISE Simulator (ISim) з використанням чіпа сімейства Spartan 3 – XC3S200. Штучний нейрон може бути зконфігурованний за розробленими методом та способами у FPGA різних серій. В таблиці 2.4 показані результати такого конфігурування.

Таблиця 2.4 – Результати моделювання штучного нейрону в FPGA різних серій

FPGA		Швидкодія,нс	Ресурс FPGA (LUT)
Серія	Speed		
Xilinx Spartan 3	-5	101.579	1089
	-4	117.805	
Xilinx Spartan 6	-2	75.361	423
	-3	70.578	
	-3N	71.687	
Xilinx Zinq	-3	25.783	420
	-2	26.968	
	-1	28.547	
Virtex 7	-1	27.952	420
	-2	24.863	
Virtex 6	-2	26.879	420
	-1	29.713	
Virtex 5	-2	49.039	991
	-1	57.75	
Virtex 4	-10	70.517	1181
	-11	61.198	
	-12	53.903	
Kintex 7	-3	23.122	420
	-2	25.265	
	-1	28.146	
Artix 7	-1	30.587	420
	-2	26.127	
	-3	26.838	

Розроблений метод та спосіб апаратної реалізації нейрона та нейронних мереж дозволяють регулювати точність обчислень при різній розрядності вхідних даних, визначати займаний ресурс FPGA в кількості вентилів логічної матриці LUTs.

Апаратна реалізація радіально-базисних нейронних мереж з такою швидкістю дозволяє їм використовувати в реальному часі обчислювальні системи в управлінні швидкодіючими об'єктами. Зокрема, такі обчислювачі можуть бути використані для автопілотів в безпілотних літальних апаратах,

де вони можуть вирішити проблеми розпізнавання, керування, наближення та класифікації отриманих даних від відеокамер та інших датчиків.

2.4 Особливості реалізації нейронних мереж на FPGA за розробленими методом і алгоритмами

Однією з головних особливостей нейромереж є паралельна обробка сигналів. Багатошарові нейронні мережі представляють собою однорідну обчислювальну середу. По термінології нейроінформатики це універсальні паралельні обчислювальні структури, призначенні для вирішення самих різних класів задач. Розглянемо основні принципи паралельної обробки сигналів на FPGA та концепцію реалізації нейронних мереж на FPGA.

Обробка за допомогою конвеєра – це метод, при якому кілька інструкцій або пакетів даних можуть оброблятися процесором паралельно. Для цього вся процедура обробки однієї команди розбивається на кілька етапів, по завершенні кожного з яких результат виконання зберігається в тимчасових регістрах, за допомогою яких окремі етапи і комутують між собою.

Розгортання внутрішніх циклів є відомим методом, який у ряді випадків може привести до значного зростання продуктивності. Суть методу полягає в поданні циклічних алгоритмів обробки з кінцевим числом ітерацій у вигляді довгої комбінаційної ланцюжка, що обробляється за один такт.

При багатопроцесорній обробці більше ніж один процесор в системі робить обробку вступників команд, тим самим дозволяючи здійснювати повністю паралельну обробку.

При програмно-апаратній реалізації штучних нейронних мереж на FPGA, кожний шар мережі працює паралельно іншому, тут використаний принцип конвеєра. Нейрони в кожному шарі також працюють паралельно за принципом багатопроцесорної обробки даних. Тобто кожний штучний нейрон мережі є окремим процесором і обробка інформації в кожному нейроні проходить одночасно.

Кожний нейрон представляється окремим блоком, як показано на Рис. 2.12, який в свою чергу складається з декількох паралельних процесів, а нейронна мережа багатопроцесорною системою. Мова програмування дозволяє явно вказувати сигнали, які запускають процес. Для запуску нейрона використовується вхідний сигнал цього нейрона.

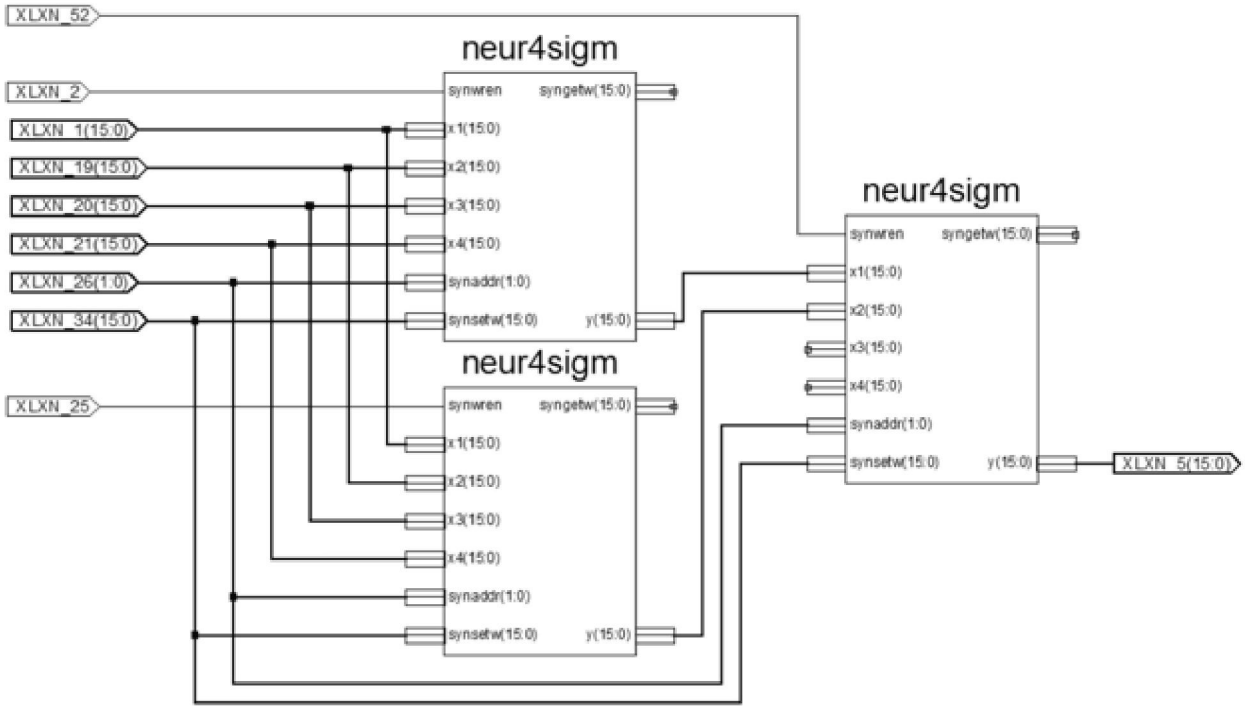


Рис. 2.12 – Нейронна мережа побудована на FPGA

Було промодельовано серію нейроних мереж прямого розповсюдження, займаний ними ресурс FPGA в кількості вентилів логічної матриці LUTs, швидкодія та похибка представленні в таблиці 2.5.

Таблиця 2.5 – Результати моделювання нейронних мереж в FPGA

Нейронна мережа	Кількість синапсів	Ресурс FPGA (LUT)	Вихідне значення ШНМ		Похибка	Швидкодія, нс
			MatLab	FPGA		
1-1-1	3	484	0.56389366	0.546875	0.017019	120.243
1-2-1	5	585	0.62573413	0.609375	0.016359	126.216
1-3-2	10	723	0.68372392	0.671875	0.011849	127.772

1-5-1	11	523	0.78328235	0.765625	0.017657	128.754
2-2-1	8	488	0.63863534	0.625	0.013635	150.481
2-3-1	11	790	0.70144096	0.6875	0.013941	130.580
2-4-2	18	899	0.75747616	0.75	0.007476	133.601
4-4-4	36	1104	0.79223947	0.765625	0.026614	133.534
3-5-1	23	725	0.82549114	0.796875	0.028616	134.047
2-4-1	14	933	0.75747616	0.75	0.007476	133.537
1-3-1	7	723	0.68372392	0.671875	0.011849	127.707
2-1-1	5	602	0.57070375	0.5625	0.008204	126.216
1-4-1	9	829	0.73651211	0.734375	0.002137	127.632
1-6-1	13	1065	0.82373748	0.796875	0.026862	132.262
3-1-1	7	723	0.577080667	0.5625	0.014581	128.705
3-2-1	11	819	0.65058263	0.625	0.025583	134.138
2-8-2	34	1369	0.907020339	0.890625	0.016395	163.237

В таблиці структура досліджуваних нейронних мереж представлена трьома числами, де перше – кількість нейронів в вхідному шарі, друге – кількість нейронів в прихованому шарі і третє – кількість нейронів в вихідному шарі. Отримані значення в LUTs можуть дещо змінюватись при зміні розрядності синаптичних ваг нейронів. В порівнянні з [120] аналогом де представлена ШНМ 3-2-1, що займає 1125 LUT, реалізована в цій роботі мережа займає 819 LUT.

Для отримання похибки апаратно реалізованої ШНМ в FPGA, зібрано перевірочну модель в програмному пакеті MatLab, Рис. 2.13, в таблиці 2.5 представленні значення виходу ШНМ реалізованої на FPGA, та в MatLab, при однакових входах ШНМ та параметрах, їх різниця в колонці «Похибка».

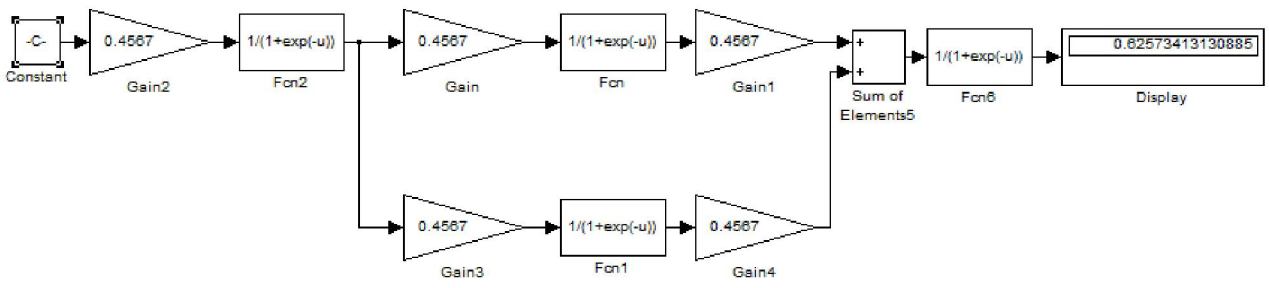


Рис. 2.13 – Модель нейронної мережі 1-2-1 в MatLab

Апаратна реалізація нейронних мереж Хопфілда на FPGA відрізняється від апаратної реалізації багатошарових нейронних мереж прямого розповсюдження введенням додаткових зворотних зв'язків і елементів затримки в часі на цих зв'язках, що в свою чергу займає більше ресурсу FPGA. Моделювання нейронних мереж Хопфілда з одним шаром і трьома нейронами в ньому та двома шарами по три нейрони в кожному показало, що для їх реалізації необхідний ресурс FPGA становить відповідно 2521 і 4945 LUTs при реалізації на чіпі Xilinx Spartan 3.

В роботі були апаратно реалізовані RBF-мережі. На Рис. 2.14 представлена RBF нейронна мережа, топології 2-3, побудована на FPGA.

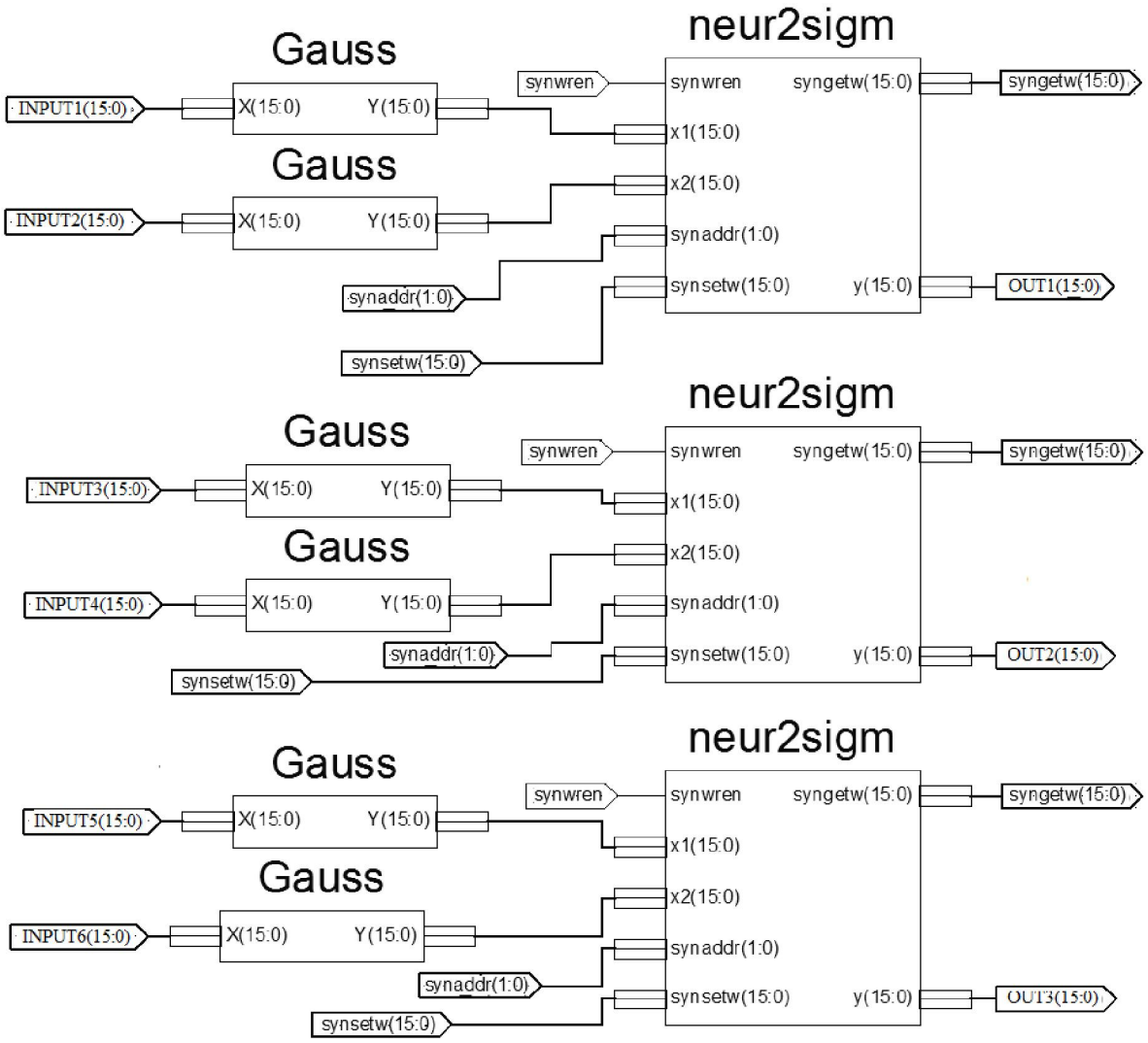


Рис. 2.14 – RBF нейронна мережа 2-3 побудована на FPGA

Результати моделювання RBF-мережі і відповідно займаний ними ресурс FPGA – LUTs, швидкодія представленні в таблиці 2.6. Топологія нейронних мереж представлена двома числами, де перше кількість функцій Гаусса в вхідному шарі, друге кількість нейронів з сигмоїдальною функцією активації в вихідному шарі. При конфігуруванні радіально-базисних нейронних мереж використано чіп сімейства Spartan 3. Отримані значення в LUTs можуть дещо змінюватись при зміні констант, які задають синаптичні ваги нейронів.

Таблиця 2.6 – Результати моделювання нейронних мереж в FPGA

Нейронна мережа	Кількість функцій Гауса	Ресурс FPGA (LUT)	Швидкодія, нс
1-1	1	452	86.671
2-1	2	771	86.736
3-1	3	1090	86.761
4-1	4	1409	86.770
1-2	2	695	90.043
2-2	4	1110	90.107
3-2	6	1525	90.133
4-2	8	1940	90.141
1-3	3	942	93.913
2-3	6	1470	93.978
3-3	9	1998	94.003
4-3	12	2526	94.012
1-4	4	1193	101.579
2-4	8	1838	101.755
3-4	12	2483	101.78
4-4	16	3128	101.789
1-5	5	1445	105.123
2-5	10	2192	105.188
1-6	6	1679	107.54
2-6	12	2511	107.604
1-8	8	2156	112.968
2-8	16	3196	113.033

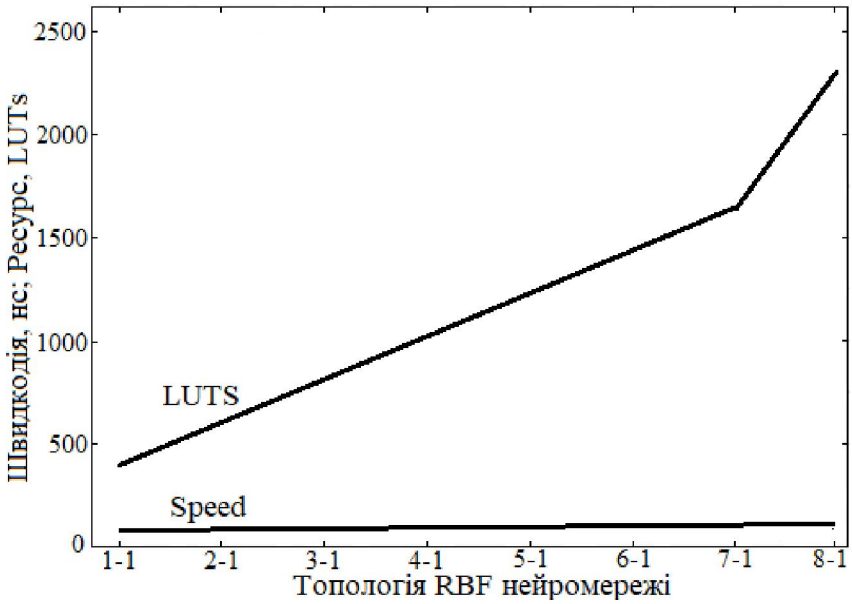


Рис. 2.15 – Графік залежності кількості використаного ресурсу від кількості нейронів та швидкодії RBF-нейромережі від кількості нейронів

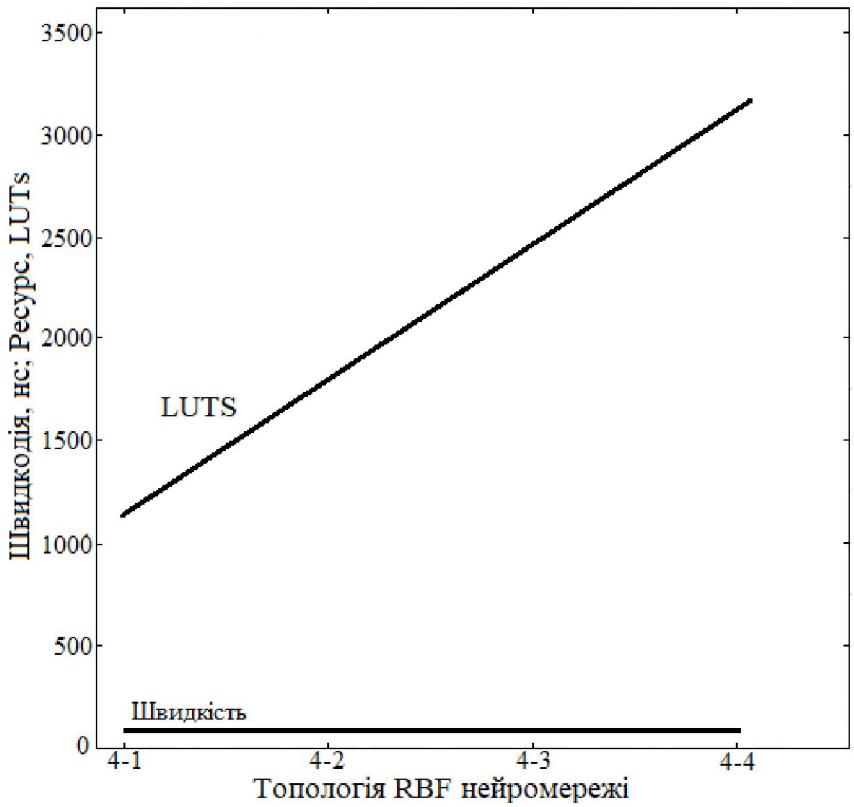


Рис. 2.16 – Графік залежності кількості використаного ресурсу від кількості нейронів та швидкодії RBF-нейромережі від кількості нейронів

На Рис. 2.15 та 2.16 представлено графіки залежності кількості використаного ресурсу від кількості нейронів та швидкодії RBF-нейромережі від кількості нейронів. Як видно з графіків при зростанні кількості нейронів в мережі, швидкодія майже не змінилась так як кожен нейрон і обчислювальні операції в ньому є окремими паралельними процесами.

У таблиці 2.7 представлено порівняння результатів моделювання нейронної мережі RBF на FPGA, реалізованій найближчим відомим аналогом [106] (Аналог) та реалізованою нейронною мережею RBF відповідно до розробленого методу та алгоритму в цій роботі (Розроблений). Моделювання виконано на однаковому чіпі Xilinx Virtex-6.

Таблиця 2.7 Порівняння результатів моделювання RBF нейронних мереж на FPGA

Нейронна мережа	Ресурс FPGA (LUT)		Швидкодія, нс	
	Аналог[106]	Розроблений	Аналог[106]	Розроблений
2-1	1170	257	41.26	27.524
4-1	2370	420	46.87	27.752

Як видно з таблиці 2.7, реалізація RBF нейронних мереж за розробленим методом та алгоритмом має більш високу швидкість і потребує меншої кількості ресурсу чіпу . У реалізованих нейронних мережах у цій роботі використовуються нелінійні, сигмоїдальні функції вихідного шару на відміну від лінійних у роботі [106].

2.5 Висновки до розділу 2

В даному розділі запропоновано метод проєктування нелінійних функцій активації штучного нейрону на FPGA. На базі запропонованого методу розроблено алгоритми апаратної реалізації штучного нейрона з сигмоїдальною фукцією активації та нейрона прихованого шару RBF-мережі з функцією активації Гауса. Проведено дослідження реалізованих штучних нейронів та ШНМ. Показано, що за рахунок розробленого методу та алгоритмів забезпечується значна оптимізація використаного ресурсу, збільшується швидкість обчислень апаратних блоків з ШНМ та їх точність в порівнянні з аналогами.

Даний метод та алгоритми можуть бути базовими для подальшого проєктування нейромережевих компонентів систем керування динамічними об'єктами на основі FPGA. До основних результатів, отриманих в даному розділі, слід віднести наступне:

– розроблено метод проєктування нелінійних функцій активації штучного нейрону на програмованих логічних інтегральних схемах, який відрізняється тим, що коефіцієнти шматково-лінійної апроксимації функції активації зберігаються у пам’яті тільки для позитивних або тільки для негативних значень аргументу, що дозволило оптимізувати кількість використаного обчислювального ресурсу та збільшити швидкодію обчислень нейронні мережі;

– розроблено алгоритм апаратної реалізації штучного нейрона з сигмоїдальною функцією активації, що дозволило підвищити швидкодію, за рахунок розпаралелення великої кількості операцій, та оптимально використати займаний ресурс в FPGA;

– розроблено алгоритм апаратної реалізації нейрона прихованого шару RBF-мережі з функцією активації Гауса за розробленим методом проєктування функції активації, що дозволило оптимально використати займаний ресурс в FPGA та підвищити швидкість синтезу таких компонентів;

– реалізовані та змодельовані штучні нейрони та нейромережі, за розробленими методом та алгоритм, за результатами моделювання показали високу точність та швидкодію.

РОЗДІЛ 3. МЕТОДИ ПРОЄКТУВАННЯ АПАРАТНИХ КОМПОНЕНТІВ НЕЙРОМЕРЕЖЕВИХ СИСТЕМ КЕРУВАННЯ НА FPGA

3.1 Розробка технології багатоетапної процедури ідентифікації складних динамічних об'єктів з використанням ШНМ

Як зазначалося в першому розділі однією з основних проблем при використанні нейронних мереж в системах керування є відсутня загальна строга теорія по вибору типу і архітектури ШНМ, що приводить до необхідності пошукового підходу чи використання алгоритмів самоорганізації, що також потребує часових затрат.

При апаратній реалізації нейромережевих компонентів на основі FPGA необхідно використовувати ШНМ мінімальної структури, які задовільняють задану точність, для оптимізації використаного ресурсу в FPGA та часових затрат на обробку інформації таким компонентом.

Під нейромережевими елементами мінімальної структури в загальному випадку розуміється багатошарова ШНМ з одним або двома прихованими шарами. ШНМ з одним прихованим шаром може апроксимувати функції з заданою точністю за відсутності обмежень на кількість нейронів в шарі. А мережі з двома прихованими шарами забезпечують велику точність при обмеженій кількості нейронів в шарах, в порівнянні з мережами з одним прихованим шаром.

При використанні нейронних мереж для реалізації процедури ідентифікації динамічних систем необхідно визначити «зовнішню» і «внутрішню» структури нейронної мережі. «Зовнішня» структура нейромережевої моделі повністю визначається вектором входу і вектором виходу. При виборі «внутрішньої» структури нейромережевої моделі необхідно визначити:

- кількість прихованих шарів;
- кількість нейронів у кожному схованому шарі;
- вид активаційної функції для кожного шару.

Збільшення числа нейронів у схованому шарі і збільшення числа прихованих шарів підвищують репрезентативні можливості нейронної мережі, але призводять до значних труднощів при практичній реалізації, збільшення часових витрат як на навчання, так і на роботу в режимі прогнозування. Це і пояснює факт прагнення використовувати мінімальну реалізацію БНМПР в більшості технічних додатків. На Рис. 3.1 представлена структура БНМПР.

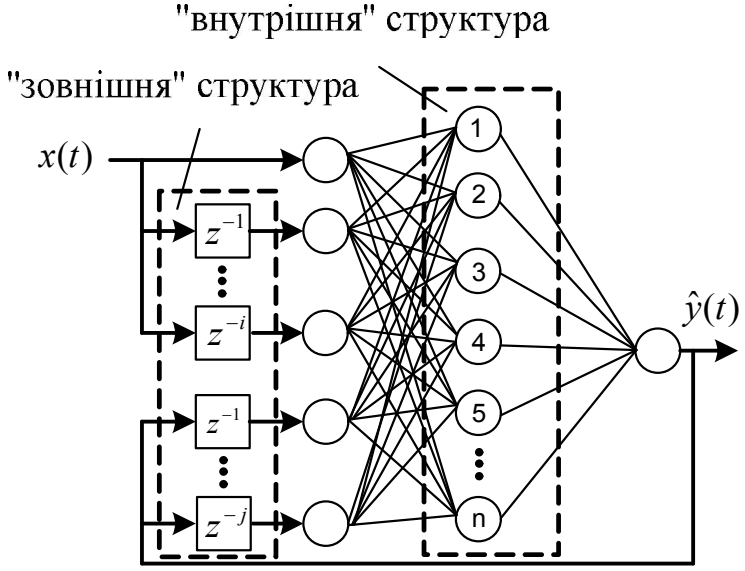


Рис. 3.1 – Структурна схема БНМПР

Виходячи з вищевикладеного, завдання побудови нейромережевої моделі зводитися до визначення «зовнішньої» і «внутрішньої» структури, а саме визначення кількості затриманих входів *i*, затриманих сигналів виходу мережі *j*, і кількості нейронів прихованого шару *n*.

Основна ідея полягає в побудові оптимальної, в за деяким критерієм, моделі за результатами спостережень над вхідними та вихідними змінними системи. На практиці реалізація процедури ідентифікації вимагає вирішення цілого ряду допоміжних завдань, основними з яких є:

- планування/проведення експерименту і попередня обробка експериментальних даних;

- вибір модельної структури;

- оцінка (оптимізація параметрів) моделі;

- прийняття рішення про адекватність моделі.

Загальна схема реалізації процедури ідентифікації, побудови моделей об’єктів та їх оцінки представлена на Рис. 3.2.

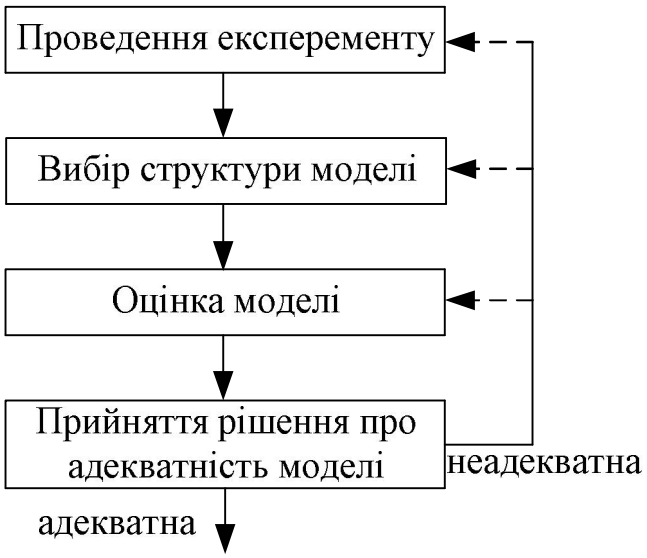


Рис. 3.2 – Загальна схема реалізації процедури ідентифікації

На Рис. 3.3 представлений схема побудови та дослідження нейромережевих моделей об’єктів керування, що пропонується в даній дисертаційній роботі.

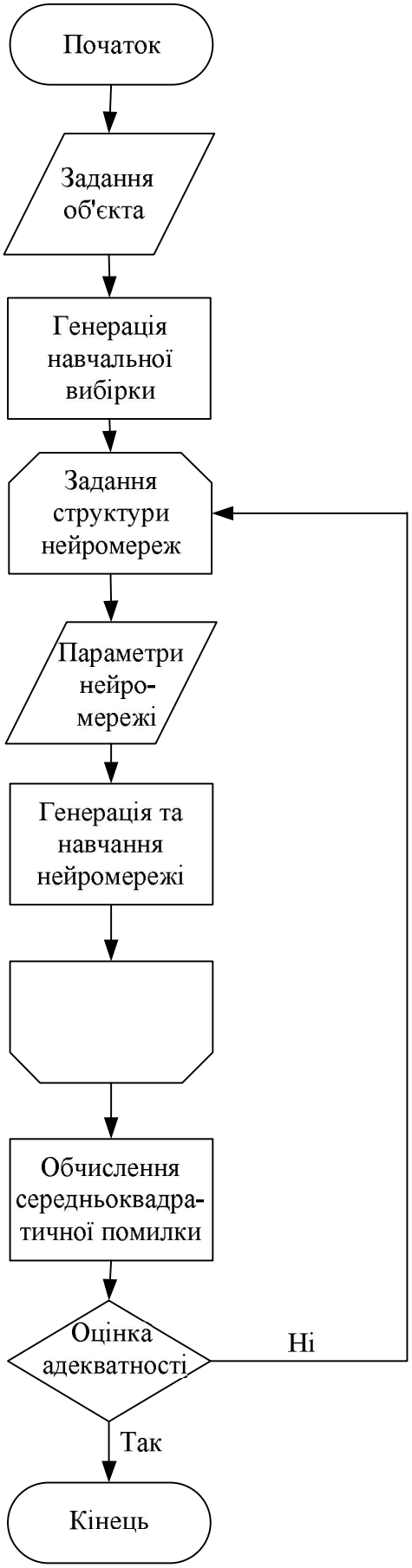


Рис. 3.3 – Схема дослідження нейромережевих моделей керування

Запропонуємо наступну технологію дослідження нейромережевих моделей об'єктів керування.

Крок 1. Обрати об'єкт для побудови його моделі на базі ШНМ.

Крок 2. Генерація навчальної вибірки. Задати параметри необхідні для генерації навчальної вибірки:

1. Задати діапазон зміни входу ($U_{\max}$, $U_{\min}$) і виходу ($Y_{\max}$, $Y_{\min}$) ОК. При виборі $Y_{\max}$, не потрібно обмежувати вихід об'єкта керування, тобто $Y_{\max}$ вибирати великим усталеного значення при $U_{\max}$.

2. Розмір навчальної вибірки слід вибирати таким чином, щоб вона містила всі можливі комбінації амплітуд і частот з робочого діапазону системи.

3. Інтервал визначає мінімальну і максимальну тривалість тестового сигналу. Причому максимальне значення повинно бути не менше часу встановлення перехідного процесу ОК при стрибкоподібному впливі.

Крок 3. Задання структури нейронної мережі прямого розповсюдження та її навчання. Задаємо розмір прихованого шару, вказуємо кількість нейронів *n*, кількість затриманих входів *i*, кількість затриманих виходів, значення *j,* алгоритм навчання і кількість навчальних ітерацій. Проводимо навчання нейронної мережі. Результатом виконання процедури навчання буде отримана нейронна мережа, яка являє собою нейромережеву модель ОК.

Крок 4. Обчислення сумарної середньоквадратичної похибки згенерованої нейронної мережі, яка математично описується наступною формулою:

$$\varepsilon = \sqrt{\frac{\sum_{i=1}^{z}(\hat{y} - y)^2}{z}}, \qquad (3.1)$$

де z – кількість значень похибки (вибирати рівне кількості елементів вектора *tout* з workspace Matlab); y – вихід об'єкта або його моделі; $\hat{y}$ – вихід нейромережевої моделі об'єкта.

Крок 5. Повторюємо крок 2-3 і створюємо необхідну кількість нейронних мереж з області можливих модельних структур з різними параметрами

затриманих входів та виходів, а також різною кількістю нейронів у прихованому шарі.

Крок 6. Обчислити загальну сумарну середньоквадратичну помилку всіх нейронних мереж з області можливих модельних структур з різними параметрами затриманих входів та виходів за формулою (3.1), а також різною кількістю нейронів у прихованому шарі.

Крок 7. Оцінити адекватність нейромережевих моделей за величиною похибки для подальшої їх апаратно-програмної реалізації. При відсутності адекватних моделей повернулись на **Крок 3** та провести нову серію дослідів.

За даною технологією можливо вирішити задачу побудови нейромережевої моделі, визначити «зовнішню» і «внутрішню» структури, а саме визначити кількість затриманих входів *i*, затриманих сигналів виходу мережі *j*, і кількості нейронів прихованого шару *n*. Розроблена технологія відрізняється від існуючих [35] етапом створення сукупності нейромережевих моделей та їх оцінювання, що дозволяє обрати мінімальну структуру нейромережевої моделі для подальшої реалізації та підвищити рівень автоматизації процедури ідентифікації.

Для реалізації даного алгоритму було розроблено програмний пакет «MIMO-Plant» в середовищі MatLab, на Рис. 3.4 зображено головне вікно програмного пакету «MIMO-Plant». Приведемо приклад дослідження нейромережевих моделей об'єкта керування за розробленою технологією на прикладі об'єкта, який має 3 вхідних та 4 вихідних параметри.

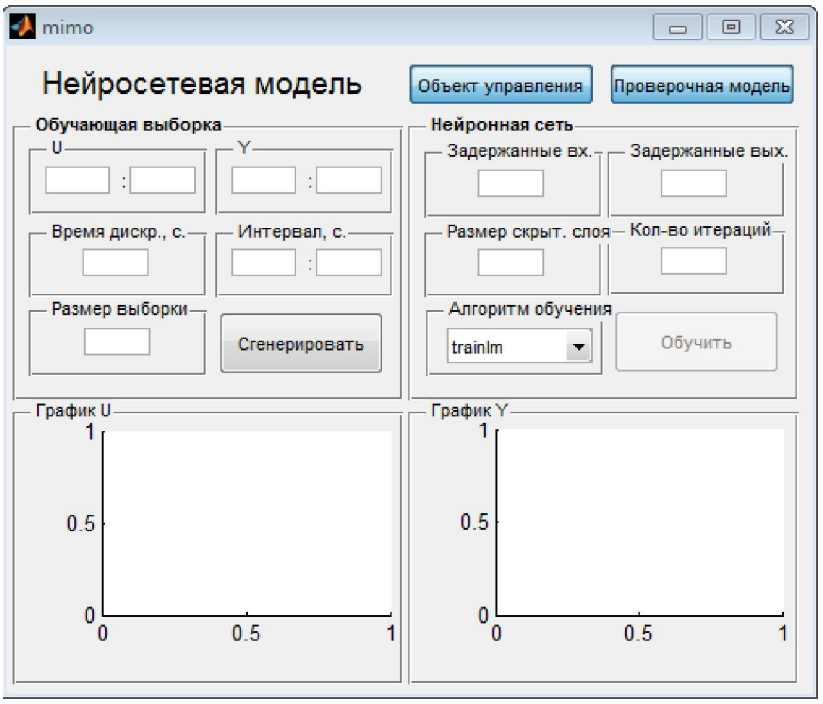


Рис. 3.4 – Головне вікно програмного пакету «MIMO-Plant»

1. Задати об'єкт керування, що моделюється. При натисканні на кнопку «Объект управления» відкриється модель об'єкта керування, Рис. 3.5.

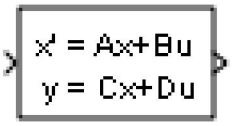


Рис. 3.5 – Математична модель об'єкта керування

Завдання об'єкта відбувається за допомогою блоку State-Space. Цей блок реалізує систему, поведінку якої можна визначити як:

$$\dot{x} = \mathbf{A}x + \mathbf{B}u$$
$$y = \mathbf{C}x + \mathbf{D}u,$$

де x – вектор станів, u – вхідний вектор, а y є вихідним вектором.

Матриця коефіцієнтів повинна мати такі характеристики:

- матриця **A** повинна бути n на n матриця, де n-число станів;
- матриця **B** повинна бути n на m матриця, де m-число входів;
- матриця **C** повинна бути r на n матриця, де r-число виходів;
- матриця **D** повинна бути r на m матрицю.

Ширина вхідного вектора залежить від кількості стовпців в матрицях **B** і **D**. Ширина вихідного вектора залежить від кількості рядків у матрицях **C** і **D**.

$$\mathbf{A}=\begin{pmatrix} -1.23 & 0 & 0 \\ 2.5 & -0.56 & 0 \\ 0 & 3 & -2.3 \end{pmatrix},\quad \mathbf{B}=\begin{pmatrix} 0.2 & 1 & 0.3 \\ 0 & 0.4 & 0.1 \\ 0 & 0.2 & -1 \end{pmatrix},$$

$$\mathbf{C}=\begin{pmatrix} 0 & 0 & 0.3 \\ 1 & 0 & 0.2 \\ 0.1 & 0.2 & 0.3 \\ 0.2 & 0 & 0.3 \end{pmatrix},\quad \mathbf{D}=\begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Для того, щоб задати об'єкт керування, вводимо матриці A,B,C,D як параметри у вікно блоку «State-space», який визивається при натисканні кнопки «Объект управления» головного меню програми.

2. Підготовчим етапом створення нейромережі є генерація навчальної вибірки, яка відбувається при натисканні кнопки «Сгенерировать», що представлена на Рис. 3.6.

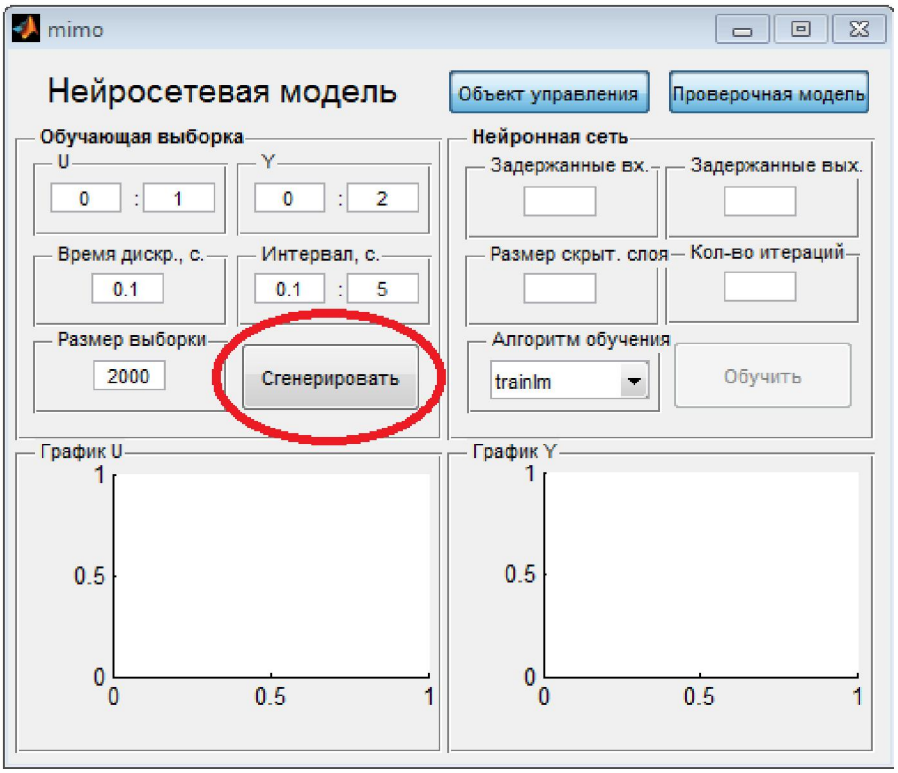


Рис. 3.6 – Генерація навчальної вибірки

Вводимо необхідні параметри для генерації навчальної вибірки:

1) Задаємо діапазон зміни входу від 0 до 1 та діапазон зміни виходу від 0 до 2.7.

2) Обираємо час дискретизації 0.1 с.

3) Обираємо інтервал від 0.1с до 5с.

4) Обираємо розмір навчальної вибірки 2000.

5) Натискаємо кнопку «Згенерувати».

3. Створюємо та навчаємо нейронну мережу. Для цього заповнюємо поля блоку «Нейронная сеть» головного меню програми, а саме:

1) У полі «Розмер скритого слоя» вказуємо кількість нейронів *n*.

2) У полі «Количество задержаних входов» вводимо значення *i.*

3) У полі «Количество задержаних выходов» вводимо значення *j*.

4) У полі «Алгоритм навчання» обираємо алгоритм навчання (trainlm) і кількість навчальних ітерацій. Нажимаємо кнопку «Обучить», як зображено на рисунку 3.7.

Якщо всі поля групи «Навчальна вибірка» заповнені, то в області *PlantInput* і *PlantOutput* повинні відображатися вид вхідного впливу на ОК та реакція ОК на вхідні дії. На Рис. 3.7 представлені графіки вхідних та вихідних сигналів та старт процедури навчання.

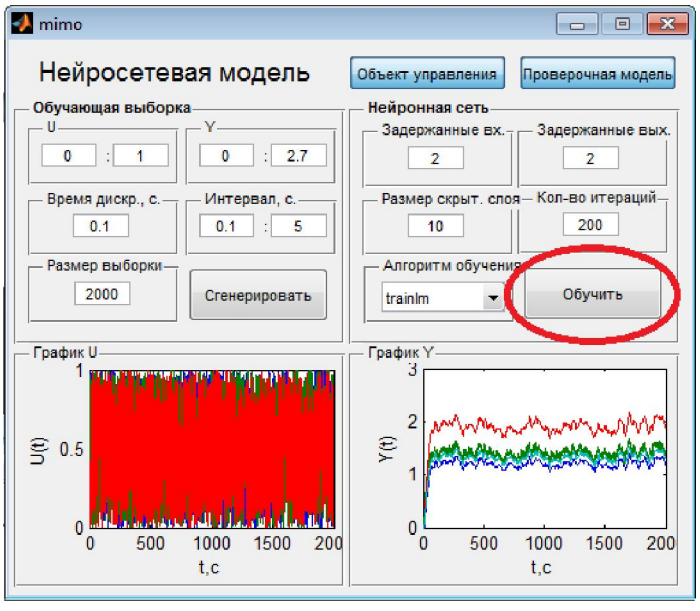


Рис. 3.7 – Навчання нейромережевих моделей об'єктів керування

Результатом виконання процедури навчання буде згенерована нейронна мережа, яка являє собою нейромережеву модель ОК, що представлена на-Рис. 3.8.

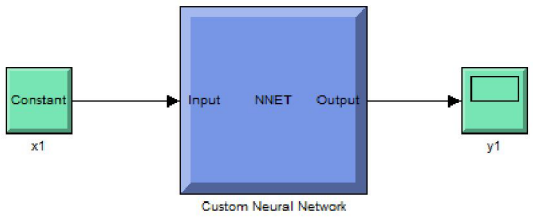


Рис. 3.8 – Згенерована нейронна мережа

4. Після того, як нейронна мережа створена та навчена, підставляємо її у модель обчислення сумарної середньоквадратичної помилки. У блок «State-space» цієї моделі задаємо об'єкт. В блок «Uniform Noice Generator» задаємо матрицю нижнього значення вхідного сигналу([0 0 0]) та матрицю верхнього значення вхідного сигналу ([1 1 1]). Кількість параметрів генератора залежить від обраного об'єкту керування. В даному випадку, оскільки маємо об'єкт керування з 3 вхідними параметрами, записуємо відповідно наведені вище параметри блоку «Uniform Noice Generator».

Для обчислення середньоквадратичної похибки згенерованої нейронної мережі, необхідно скористатися перевірочною моделлю, яка визивається при натисканні кнопки «Перевірочна модель», на Рис. 3.9 представлена модель системи в MatLab для оцінки адекватності нейромережевої моделі.

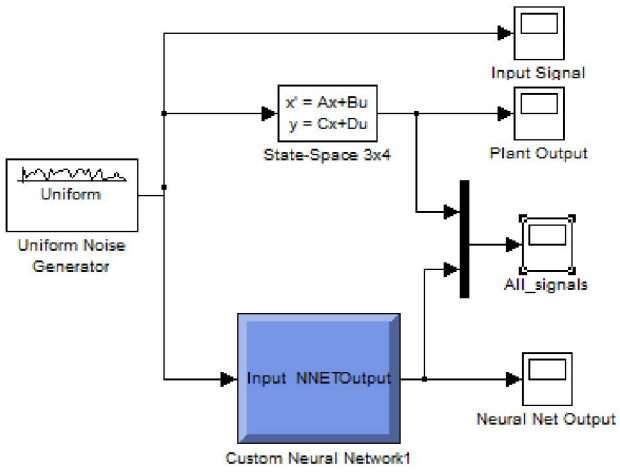


Рис. 3.9 – Перевірочна модель

Нейронна мережа «Custom Neural Network1», яка зображена на Рис. 3.9, отримана після навчання нейронної мережі та підставлена у перевірочну модель.

При натисканні на блок «All_signals» перевірочної моделі буде отримано графічне зображення виходу нейронної мережі та модельованого об'єкта керування, на рисунку 3.10 можна візуально порівняти отримані результати. На Рис. 3.10 графіки: *Y*1*ОК*, *Y*2*ОК*, *Y*3*ОК*, *Y*4*ОК* – виходи об’єкта, що моделюється; *Y*1*нм*, *Y*2*нм*, *Y*3*нм*, *Y*4*нм* – виходи нейромережевої моделі.

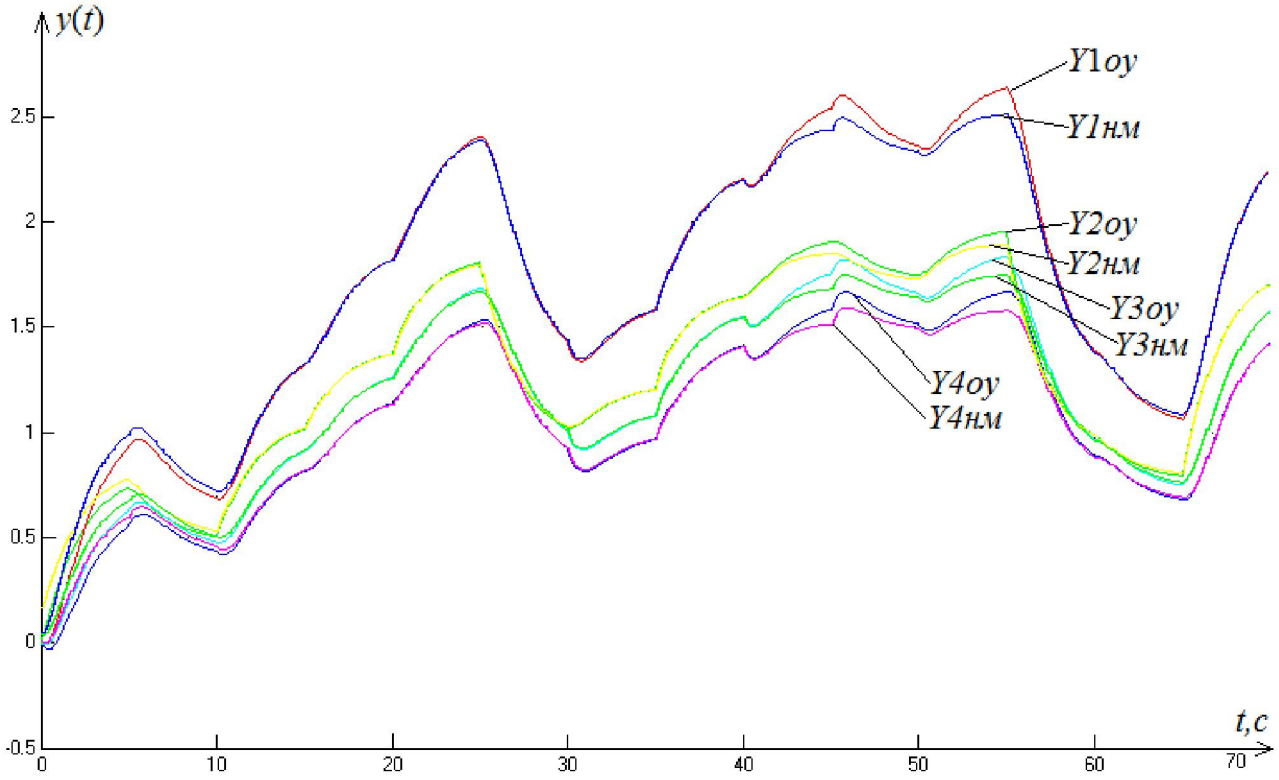


Рис. 3.10 – Графічне зображення виходу нейронної мережі та модельованого ОК

Для обчислення сумарної середньоквадратичної помилки потрібно скористатися розробленою моделлю, де:

1) На вхід подається вхідний сигнал генератором сигналу.

2) Об'єкт керування задається блоком State-Space.

3) Під об'єктом керування знаходяться 16 варіацій структури нейромережі.

4) У блоці «Display» відображається сумарна середньоквадратична помилка.

5. Повторюємо крок 2-3 і створюємо необхідну кількість нейронних мереж з області можливих модельних структур. Для цього варіюємо патаметри поля «Розмер скритого слоя» від 5 до 16 нейронів, поля «Количество задержаних входов» від 0 до 2 входів, а також поля «Количество задержаних выходов» від 1 до 3.

6. Після того, як всі нейронні мережі створені, навчені та підставлені у модель обчислення сумарної середньоквадратичної помилки, запустити модель де буде отримано значення середньоквадратичної помилку у блоці «Display»

Запустити модель обчислення загальної сумарної середньоквадратичної помилки. Дана модель зображена на Рис. 3.11.

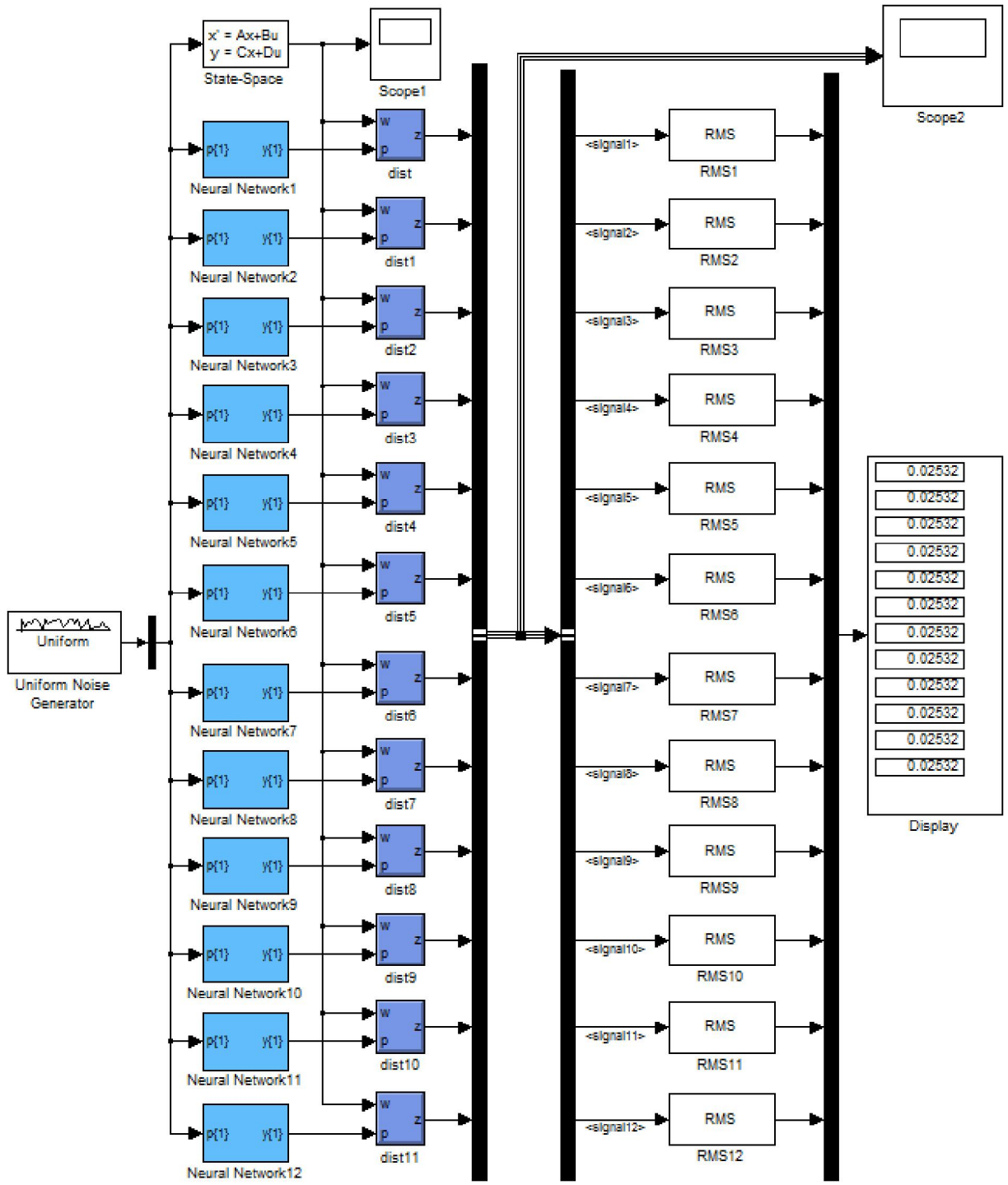


Рис. 3.11 – Модель обчислення сумарної середньоквадратичної помилки

В таблиці 3.1 представлені езультати дослідження нейромережевих моделей MIMO-об’єкта. З таблиці 3.1 видно, що оптимальною структурою, по критерію найменьшої середньоквадратичної похибки, для даної моделі з 3 вхідними та 4 вихідними параметрами є структура з 16 нейронами у прихованому шарі, 1 затриманим входом та 2 затриманими виходами. При

цьому отримано значення середньоквадратичної помилки 0.001401. Також було досліджено різні багатовимірні об'єкти 2x2, 4x2, 4x3, 3x3. Для системи 2x2 оптимальною структурою виявилась структура з 5 нейронами у прихованому шарі,1затриманим входом та 3 затриманими виходами (5-1-3). Середньоквадратична помилка при цьому становила 0.004196. Для моделі 4x2 – структура (5-2-3) з помилкою 0.001372, моделі 4x3 – структура (16-1-2) з помилкою 0.001452, моделі 3x3 – структура (16-1-3) з помилкою 0.005575.

Таблиця 3.1 – Результати дослідження нейромережевих моделей

i j / n	0_1	0_2	0_3	1_1	1_2	1_3	2_1	2_2	2_3
5	0.593	0.6415	0.663	0.0207	0.0336	0.0190	0.0079	0.0316	0.934
6	0.634	0.565	0.656	0.012	0.0138	0.0519	0.0100	0.878	0.0430
7	0.681	0.912	0.601	0.0070	0.0291	0.0373	0.0127	0.789	0.0085
8	0.720	0.4416	0.886	0.0209	0.0274	0.0159	0.0147	0.3705	0.0672
9	0.715	0.5647	0.734	0.0127	0.0179	0.0148	0.0155	0.0224	0.927
10	0.623	0.899	0.3688	0.0090	0.0132	0.0497	0.0182	0.7073	0.956
11	0.675	0.5014	0.841	0.0313	0.0182	0.0314	0.0785	0.913	0.0078
12	0.579	0.5043	0.7661	0.0079	0.0031	0.0211	0.0047	0.0368	0.0368
13	0.658	0.936	0.856	0.0033	0.0135	0.0105	0.0186	0.3052	0.0342
14	0.783	0.3992	0.887	0.0092	0.0093	0.0189	0.0277	0.0021	0.878
15	0.981	0.731	0.734	0.0130	0.0121	0.0585	0.0206	0.0202	0.0205
16	0.837	0.6442	0.558	0.0017	0.0014	0.0038	0.0058	0.1045	0.1761

Розроблена технологія та програмне забезпечення дозволить визначити мінімальну структуру ШНМ, які задовільняють задану точність при побудові прямих та інверсних моделей об'єктів керування, для оптимізації використаного ресурсу в FPGA та часових затрат на обробку інформації таким компонентом. При використанні нейронних мереж для реалізації процедури ідентифікації та керування динамічних систем розроблений алгоритм та програмне забезпечення дозволяє визначити «зовнішню» і «внутрішню» структури нейронної мережі.

3.2 Метод проєктування прямої та інверсної моделі об'єкта керування з їх апаратною реалізацією на FPGA

На основі розроблених раніше методу, алгоритмів та технології розробимо метод проєктування прямої та оберненої нейромережевої моделі об'єкта керування, при їх апаратній реалізації на FPGA, що дозволяють реалізовувати функції ідентифікації, адаптації та керування динамічними об'єктами в реальному часі. Дані моделі можуть бути базовими для структурного синтезу функціонально більш складних систем керування. Обидві моделі використаються для обчислення векторів стану об'єкта і формування функції керування ним.

Реалізація методу проєктування прямої та оберненої нейромережевої моделі об'єкта керування, при їх апаратній реалізації на FPGA, здійснюється за вісім базових етапів: 1) проведення експерименту; 2) вибір області можливих модельних структур; 3) оцінка моделей з області «можливих» модельних структур; 4) реалізація нейромережевої моделі на FPGA; 5) оцінка «вартості» реалізації моделі на FPGA; 6) прийняття рішення про можливість практичної реалізації; 7) оптимізація структури моделі; 8) Синтез конфігурації для FPGA. Структура методу представлена на Рис. 3.12.

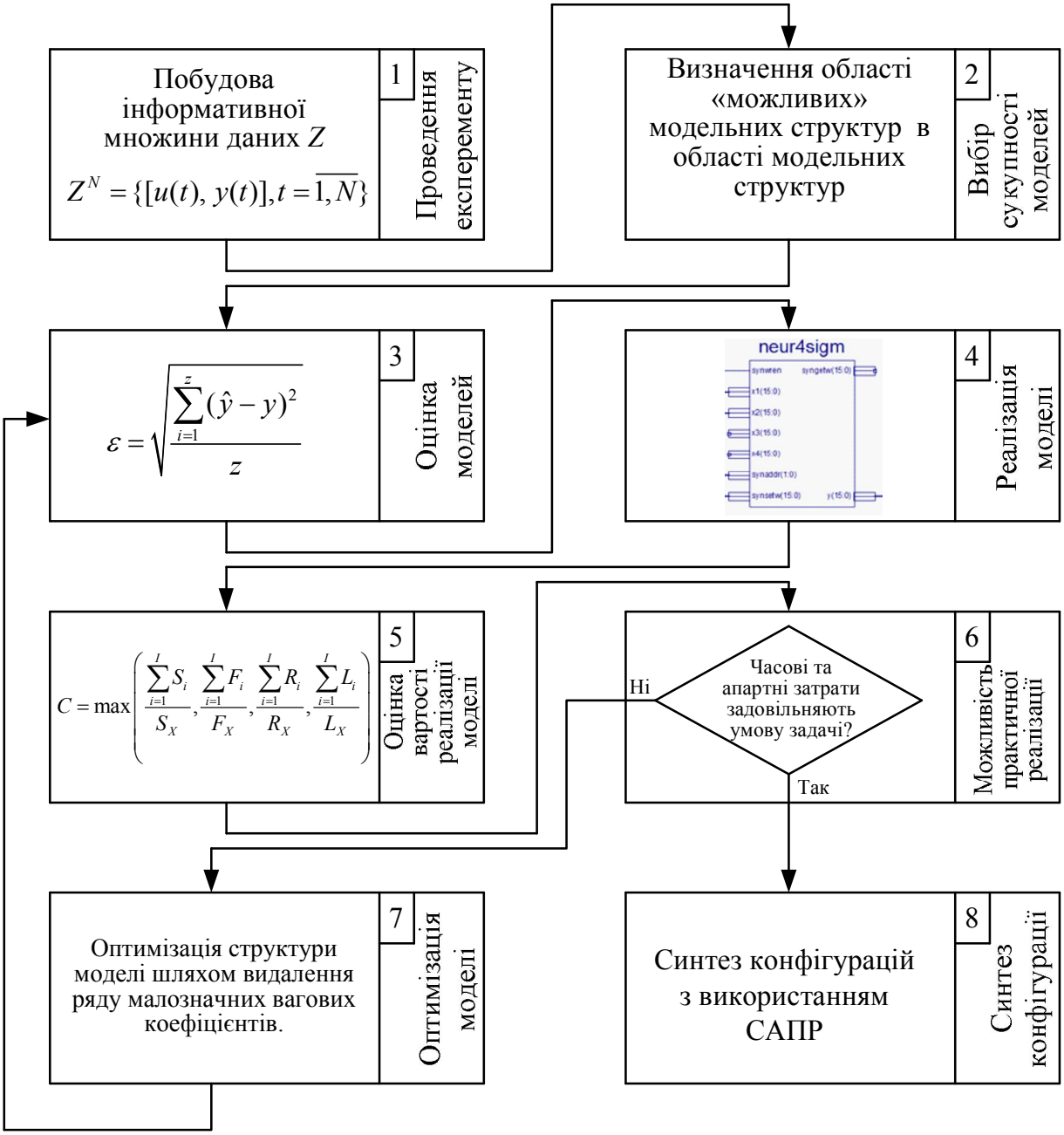


Рис. 3.12 – Структура методу проєктування прямої та оберненої нейромережевої моделі об'єкта керування при їх апаратній реалізації на FPGA

Метод проєктування нейромережевих моделей об'єкта керування при їх апаратній реалізації на FPGA виконується наступним чином.

Етап 1. Проведення експерименту.

Основним завданням проведення експерименту є побудова інформативної множини даних *Z*, придатної для побудови працездатної моделі:

$$Z^N = \{[u(t), y(t)], t = \overline{1, N}\}, \qquad (3.2)$$

де: $u(t)$ – тестовий сигнал; $y(t)$ – реакція об'єкту на тестовий сигнал; t – час; N – розмірність множини Z.

Побудова інформативної множини даних Z може бути виконана шляхом імітаційного моделювання (у випадку якщо модель об'єкту відома) або шляхом проведення серії експериментів з реальним об'єктом (у випадку, якщо модель об'єкту не відома.).

Для нелінійних об'єктів надзвичайно важливо, що б в множині експериментальних даних Z^N були представлені всі можливі комбінації амплітуд і частот з робочого діапазону системи. Один з можливих варіантів тестового сигналу, що задовольняє вказаним вимогам, описується виразом:

$$u(t) = u(t-N) + e\left(\text{int}\left[\frac{t-1}{N}\right]+1\right), t = 1,2,\ldots \tag{3.3}$$

де: $e(t)$ – білий шум з дисперсією σ_e^2.

Іншим варіантом тестового сигналу є синусоїда з наростаючою частотою ω_c і амплітудою A_c, що змінюється:

$$\begin{aligned} u(t) &= u_0 + A_c \sin(\omega_c t), \\ \omega_c &= \omega_н + \frac{(\omega_к - \omega_н)}{N}, \\ A_c &= A_н + \frac{(A_к - A_н)}{N}, \end{aligned} \tag{3.4}$$

де $\omega_н$, $\omega_к$ – відповідно початкове і кінцеве значення частоти синусоїдального сигналу; $A_п$, A_K – відповідно початкове і кінцеве значення амплітуди синусоїдального сигналу.

Для ефективного використання отриманої тестової множини експериментальних даних необхідно виконати його попередню обробку (фільтрацію, видалення надмірних даних і викидів, масштабування), метою якого є отримання найбільш значущої інформації і приведення її до певного вигляду, необхідного для навчання нейромережевої моделі.

Етап 2. Вибір області можливих модельних структур.

Під модельною структурою розумітимемо структуру нейронної мережі, яка може бути умовно розділена на «внутрішню структуру» і «зовнішню структуру».

Внутрішня структура нейронної мережі визначається: топологією мережі, кількістю прихованих шарів, числом нейронів і видом активаційних функцій в кожному шарі.

Зовнішня структура нейронної мережі визначається вектором входу $\phi(t)$ (регресором). Вектор входу $\phi(t)$ нейронної мережі можна представити як:

$$\phi(t) = [\phi_1 \ldots \phi_k]^T = [\phi_1(t-1) \ldots \phi_1(t-d_1) \ldots \phi_k(t-1) \ldots \phi_k(t-d_k)]^T, \qquad (3.4)$$

де: ϕ_k – k-а компонента регресора; d_k – «глибина» регресора.

Під вибором регресора мається на увазі визначення компонент регресора ϕ_k і глибини регресії d_k, тобто кількості d значень k-ої компоненти регресора в попередні значення часу. У якості компонент регресора зазвичай використовуються ті параметри системи (процесу), які можуть бути безпосередньо виміряні (або оцінені) в режимі функціонування. Наприклад, для одновимірних об'єктів в якості компонентів регресора використовується значення входу $u(t)$ і виходу $y(t)$ об'єкту. Вибір глибини регресії визначається динамікою об'єкта.

Таким чином, завдання вибору структури нейронної мережі зводиться до визначення «глибини» регресора $\varphi(t)$ («зовнішня» структура) і кількості нейронів прихованого шару n («внутрішня» структура) багатошарової нейронні мережі, що містить один прихований шар нейронів з сигмоїдальними функціями активації і вихідний шар з лінійними функціями активації.

Визначення області «можливих» модельних структур $V(n, \varphi_1, \ldots, \varphi_k)$ в області модельних структур $M(n, \varphi_1, \ldots, \varphi_k)$, виконується на основі аналізу апріорної інформації про об'єкт і його динаміку.

Етап 3. Оцінка моделей з області можливих модельних структур.

Для оцінки моделей з області можливих модельних структур використано комплексний формалізований підхід до реалізації багатоетапної процедури побудови нейромережевих моделей складних динамічних об'єктів, розроблено програмний пакет «MIMO-Plant» в середовищі MatLab, який описаний в розділі 3.1. Процес оцінки моделей полягає в наступному, для кожної моделі з області $V\left(n,\varphi_1,\ldots,\varphi_k\right)$ обчислюється значення критерію адекватності, у якості котрого можна використовувати значення середньоквадратичної помилки (3.1). За критерієм середньоквадратичної та складністю моделі обирається модель із області можливих модельних структур $V\left(n,\varphi_1,\ldots,\varphi_k\right)$.

Етап 4. Реалізація нейромережевої моделі на FPGA.

Для апаратної реалізації нейромережевої моделі на FPGA, обраної на третьому етапі, використовуються метод та алгоритми апаратної реалізації ШНМ розроблені та описані в другому розділі.

Етап 5. Оцінка «вартості» реалізації моделі на FPGA.

Для вибраної моделі обчислюється значення критерію «вартості» апаратної реалізації нейромережевої моделі на FPGA. Під «вартістю» реалізації C в даному випадку розуміється частина ресурсів (Slices), кількість D–тригерів (Flip Flops), об'єм вбудованої блокової пам'яті (BRAM), кількість чотиревходових таблиць перетворення (Look-up Table)), необхідних для реалізації отриманої моделі на вибраному типі кристала FPGA

$$C = \max\left(\frac{\sum_{i=1}^{I} S_i}{S_X}, \frac{\sum_{i=1}^{I} F_i}{F_X}, \frac{\sum_{i=1}^{I} R_i}{R_X}, \frac{\sum_{i=1}^{I} L_i}{L_X}\right), \tag{3.5}$$

де: S_i, F_i, R_i, L_i – кількість Slices, Flip Flops, BRAM, Look-up Table, відповідно, що необхідна для реалізації нейромережевої моделі; S_X, F_X, R_X, L_X – кількість Slices, Flip Flops, BRAM, Look-up Table, відповідно, яка міститься у вибраному кристалі FPGA; I – сумарна кількість елементів (ваги, зсуви, функції активації, затримки) тих, що реалізовують нейронну мережу.

Умовою реалізуємості нейромережевої моделі на вибраному кристалі FPGA є виконання нерівності

$$C < 1. \qquad (3.6)$$

Етап 6. Прийняття рішення про можливість практичної реалізації.

Якщо отримана модель не задовольняє якомусь критерію, то необхідно виконати попередні кроки алгоритму побудови моделі, аж до проведення нової серії експериментів.

Відповідно до даного алгоритму, якщо умова (3.6) виконується, то дану модель можна реалізувати на вибраному кристалі FPGA, перейти на етап 8, якщо ж умова (3.6) не виконується, то модель на такому кристалі FPGA реалізувати неможливо. В цьому випадку забезпечити виконання умови (3.6) можна наступним чином: виконати процедуру оптимізації структури моделі, перейти на етап 7; вибрати кристал FPGA більшої ємності і/або іншого сімейства; збільшити число кристалів FPGA.

Етап 7. Оптимізація структури моделі

Оптимізація структури нейромережевої моделі проводиться шляхом видалення ряду малозначних вагових коефіцієнтів. Для реалізації цього можуть бути використані такі алгоритми: алгоритм послідовного зменшення структури; алгоритм послідовного збільшення структури; генетичний алгоритм; Optimal Brain Damage (OBD); Optimal Brain Surgeon (OBS).

Останній алгоритм є найбільш відомим і широко поширеним. Після виконання мінімізації структури необхідно заново оцінити адекватність моделі етап 3 і виконати етапи 4, 5, 6.

Етап 8. Синтез конфігурацій.

На даному етапі засобами пакету САПР синтезуються конфігурації для прошивки FPGA.

Запропонований метод проєктування прямої та оберненої нейромережевої моделі об'єкта керування при їх апаратній реалізації на FPGA, відрізняється від існуючих етапом генерації та дослідження нейромережевих моделей, їх оцінки, та етапом реалізації нейромережевої моделі на FPGA, що до-

зволяє прискорити та автоматизувати процес створення нейромережевих компонентів систем керування, що реалізують функції ідентифікації, адаптації та керування динамічними об'єктами в реальному часі.

3.3 Метод оптимізації коефіцієнтів нейронних мереж генетичним алгоритмом при апаратній реалізації на FPGA

Розробка паралельних обчислювальних систем включає в себе всі риси розробки послідовних. Але в розробці паралельних обчислювальних систем є три додаткових чітко визначених етапи.

Перший етап – визначення паралелізму. На цьому етапі проводиться аналіз розв'язуваної задачі. Виділяються підзадачі, які можуть виконуватися паралельно.

Другий етап – організація паралелізму. На даному етапі необхідно створити або змінити структуру завдачі. Для ефективного виконання підзадач паралельно. Для цього потрібно знайти залежності між під задачами, після чого організувати вихідний код таким чином, щоб даними підзадачами можна було ефективно управляти.

Третій етап – це реалізація паралелізму. На цьому етапі відбувається реалізація паралельного алгоритму в паралельну обчислювальну систему.

Розробці паралельних обчислювальної системи навчання оптимізації вагових коефіцієнтів нейронних мереж необхідно дослідити характеристики реалізуємого алгоритму. Підготовка апаратної реалізації обрахунків процедур навчання нейронної мережі генетичним алгоритмом здійснюється на основі графа

$$GDF = (A, D),$$

де A – множина вершин, відповідних операціям; D – множина дуг, відповідних потокам даних.

На Рис. 3.13 представлено граф обрахунків генетичного алгоритму, виділені задачі, які можуть виконуватись паралельно.

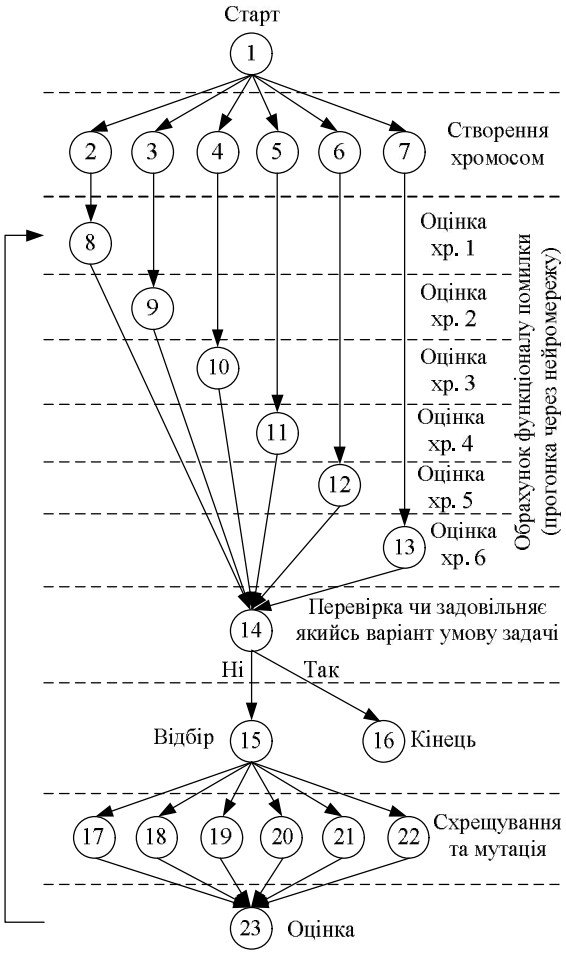


Рис. 3.13 – Граф обрахунків

Для апаратної реалізації еволюційної стратегії оптимізації синаптичних ваг нейромережі на FPGA необхідно також розробити генератор випадкових чисел. Генерація випадкових чисел проходить на основі мультипликативного конгруентного методу, відповідності з яким кожне наступне число формується на базі попереднього за формулою:

$$r_{i+1} = \mathrm{mod}\left(k \cdot r_i, M\right),$$

де M – модуль ($0 < M$), k – множник ($0 \le k < M$).

Число M повинне бути досить великим, оскільки період генерації чисел не може мати більше ніж M елементів. Логічно обрати $M = 2^N$, оскільки у цьому випадку операції ділення, які мають місце в процедурах обчислень і які для FPGA не є стандартними, можуть бути замінені операціями зсуву.

Але для генетичного алгоритму, необхідні випадкові числа в діапазоні від 0 до 7. Їх можна отримати за формулою:

$$r_i^* = \mathrm{mod}(r_i, 8).$$

Тобто для генетичного алгоритму використовуємо 3 молодші розряди згенерованого випадкового числа. Це можливо зробити, оскільки існує теорема що, при використанні конгруентного метода генерації випадкових чисел, молодші розряди отриманого випадкового числа ведуть себе так само випадково, як і старші.

Число 8, обумовлене кількістю бітів у хромосомі. Якщо змінити кількість біт, то виникне необхідність генерації випадкових чисел у іншому діапазоні. Наприклад для діапазону 1…10, це можна зробити за формулою:

$$r_i^* = \mathrm{mod}(r_i, 8) + \mathrm{mod}(r_i, 4).$$

При реалізації було обрано, $k = \frac{29}{8} = 3.625$; $M = 2^{20} = 1048576$.

Для реалізація генетичного алгоритму на FPGA для навчання нейронних мереж запропонуємо наступний метод:

Крок 1. На даному кроці відбувається початкова ініціалізація. ССтворюється ШНМ на FPGA, за методом наведеним в розділі 2, задаються початкові параметри нейромережі та кодуються в хромосому. Створюємо набір хромосом, аналогічних за розміром. Набір хромосом представлений двомірним масивом, таблиця 3.2. Кожна строчка масиву відповідає хромосомі і містить інформацію про весь набір вагових коефіцієнтів мережі.

Таблиця 3.2 – Зображення популяції хромосом за допомогою двовимірного масиву

№ Хр.	Вагові коефіцієнти							
	1-й нейрон			2-й нейрон			N-й нейрон	
	1	2	3	4	5		n-1	n

1	-1	-1	-1	-1	-1	...	-1	-1
2	-1	-1	-1	0	0	...	0	0
3	1	1	1	1	1	...	1	1
...								
k-1	0	0	0	0.5	0.5	...	-1	-1
k	0.5	0.5	0.5	0.5	0.5	...	0.5	0.5

Крок 2. Декодування виконується з використанням стандартних функцій мови VHDL, а саме *To_integer* та *Signed*. Значення фітнес-функції визначають наскільки отриманий вихід мережі відрізняється від необхідного, розраховується як різниця між необхідним виходом мережі і отриманим. Підставляємо по черзі значення вагових коефіцієнтів з кожної хромосоми та вираховуємо похибку.

Крок 3. Якщо одне з значень задовольняє умову задачі, переходимо на **крок 7**.

Крок 4. Відбір хромосом для подальшого схрещування та мутації, що досягається сортуванням за значенням фітнес-функції.

До початку сортування маємо масив хромосом *chrs* і масив значень фітнес функцій *rel*. Номери елементів у цих масивах відповідають одне одному, тобто першій хромосомі відповідає перший елемент масиву *rel*, другій – другий, і т.д. Сортування хромосом необхідно виконати за значенням фітнес-функцій. Сортуються обидва масиви, масив *rel* буде відсортовано за зростанням значень, а масив *chrs* сортується для того, щоб елементи масивів продовжували відповідати одне одному, щоб кожному значенню фітнес-функції ставилася у відповідність хромосома, з використанням якої було розраховано це значення. Таким чином при завершенні алгоритму хромосоми буде відсортовано за зменшенням їх рівня пристосованості, перша хромосома у масиві chrs матиме найкраще значення фітнес-функції, остання – найгірше.

Кроку 5 та 6. Застосувати оператори схрещування та мутації для хромосом, відібраних на попередньому кроці.

Розраховуємо: $n_2 = \frac{1+\sqrt{1+8N}}{2}$, $n_1 = n_2 - 0.5 - \sqrt{(n_2 - 0.5)^2 - 2N}$,

$$H' = \sum_{i=1}^{n_1} \sum_{j=i+1}^{n_2} \sum_{m=1}^{mMax} \left(\sum_{k=1}^{b-1} H_{imk} 2^k + H_{jmb} 2^b + \sum_{k=b+1}^{B} H_{imk} 2^k \right), \quad n_{1,2} < N$$

H' – новий набір хромосом, H – попередній. Усі хромосоми розглядаємо як набір бітів. Тоді, перша сума визначає першу хромосому для схрещення (її номер i), друга сума – другу (номер j). Вираз у дужках визначає операції з бітами нової хромосоми, де b і k – номери бітів, 2^b, 2^k – позначають позицію біта у двійковому числі, H_{ik} – k-й біт i-ї хромосоми, H_{jb} – біт з номером b у j-ї хромосомі, m – номер вагового коефіцієнту в хромосомі. Таким чином, за формулою, усі біти нової хромосоми копіюються з i-ї хромосоми попереднього покоління, окрім біта з номером b, він копіюється з j-ї хромосоми.

Дана операція реалізується за допомогою циклів ***for***. Після виконання формуємо нове покоління і переходимо на **крок 2**.

Крок 7. Зупиняємо роботу генетичного алгоритму та підставляємо в нейромережу отримані значення.

Далі розглянемо приклад навчання одного нейрону з трьома входами і відповідно трьома ваговими коефіцієнтами. Апаратна реалізація генетичного алгоритму за запропонованою методикою виконана на чіпі сімейства Spartan 3 – XC3S200 в середовищі Xilinx ISE Design Suite 13.2 та промодельована в ISE Simulator (ISim). Процес навчання представлений на Рис. 3.13.

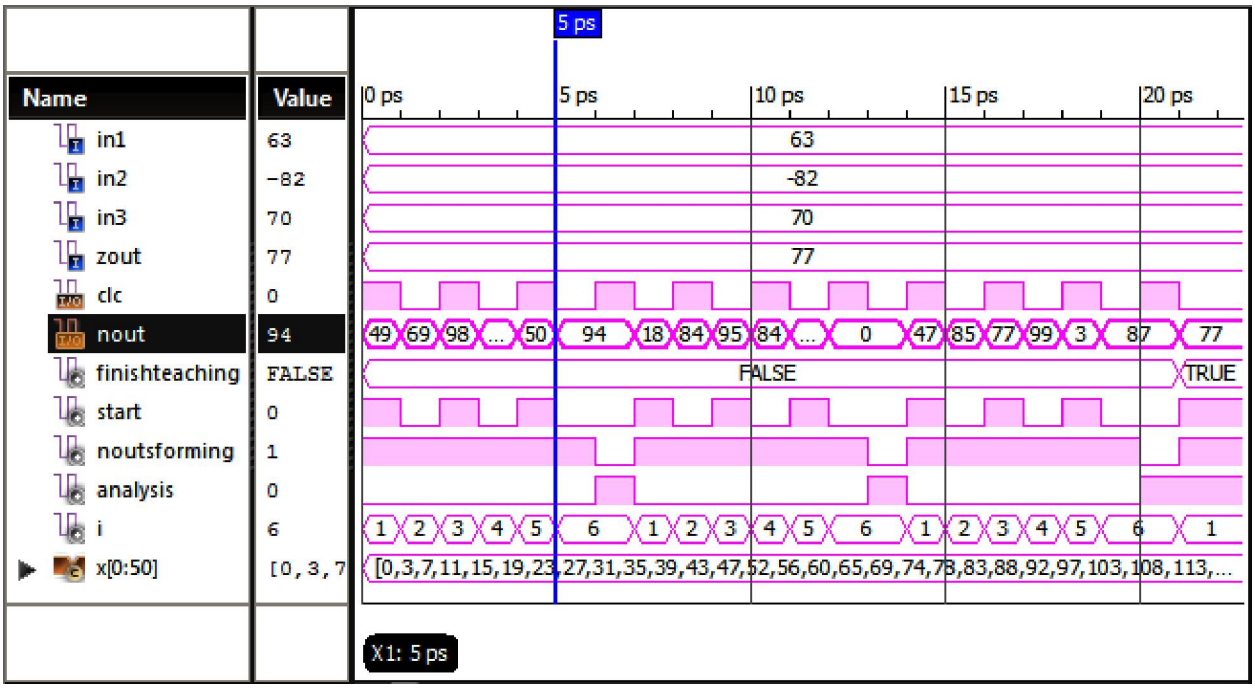


Рис. 3.13 – Процес навчання нейронної мережі генетичним алгоритмом

Сигнали *in*1, *in*2, *in*3, *zout* – подаються на входи, *in*1, *in*2, *in*3 – входи нейрона, *zout* – вихід, який повинен бути досягнутий в результаті навчання. *clc* – сигнал, що формує затримки в алгоритмі навчання. *Nout* – вихід нейрона на кожному кроці навчання ("..."– відповідає 100, оскільки, тризначним числом не вистачає місця для відображення, а 100 – єдине тризначне число, яке можна отримати на виході). *FinishTeching* – сигнал, який зберігає інформацію про те що навчання мережі завершено. *Start* – сигнал, що запускає процес, що формує вихід мережі для кожного нового набору вагових коефіцієнтів. Змінює своє значення при запуску цього процесу. *i* – номер хромосоми, на основі якої були сформовані вагові коефіцієнти, перед запуском процесу, що формує вихід нейронної мережі (перед установкою сигналу *start* в 1) *NoutsForming* – сигнал, який встановлений в 1 поки відбувається прогонка хромосом через нейронну мережу. Переходить з 0 в 1 коли всі 6 хромосом оброблені, коли можна переходити до аналізу результатів: завершити алгоритм або за допомогою схрещування і мутації сформувати нові хромосоми. Перехід сигналу з 0 у 1 говорить про початок нової ітерації генетичного алгоритму. Рішення про перехід до аналізу даних, отриманих при прогоні через

нейронну мережу, і рішення про початок нової ітерації в алгоритмі навчання, приймається в двох різних процесах. Мова VHDL не допускає зміни значення одного сигналу в двох різних процесах, тому введено додатковий сигнал – *analysis*. Після «прогону» 6-ти хромосом через нейронну мережу значення цього сигналу переходить з 0 в 1, а при початку нової ітерації генетичного алгоритму – з 1 в 0. При переході сигналу *analysis* з 1 в 0, змінюється і *NoutsForming*, але в іншому процесі. *Analysis* – сигнал, який встановлений в 1, поки відбувається аналіз інформації, отриманої при прогоні хромосом через нейронну мережу. Сигнал *clc* формує затримки в часі в 1 пікосекунду. Перші 6 пікосекунд відбувається «прогонка» хромосом через нейронну мережу (сигнал *NoutsForming* = 1). У цей час значення сигналу і змінюється від 1 до 6 кожну пікосекунду, на початку кожної пікосекунди змінюється значення сигналу *Start* і формується вихід нейронної мережі для *i*-ї хромосоми (змінюється значення сигналу *Nout*). На 7-й пікосекунді відбувається аналіз інформації, отриманої при прогоні хромосом через нейронну мережу (сигнал *Analysis* = 1) Оскільки, на цій ітерації не досягнуто необхідне значення виходу (на протязі 1-6 пікосекунди *Nout* не дорівнювало *Zout*), формуються нові хромосоми і запускається нова ітерація генетичного алгоритму. Протягом 8-14 пікосекунд сигнали *i*, *start*, *NoutsForming*, *Analysis*, змінюються точно так само як і протягом попередніх 7-ми пікосекунд, проте значення сигналу *Nout* інші, оскільки алгоритм виконується для інших хромосом. На третій ітерації третьої хромосоми дає необхідне значення виходу мережі – 77, тому після аналізу на 21-й пікосекунді сигнал *FinishTeaching* приймає значення *true*, алгоритм навчання завершується, сигнали *i*, *start*, *NoutsForming*, *Analysis* перестають міняти свої значення, вагові коефіцієнти мережі приймають значення відповідні 3-й хромосомі на останній ітерації алгоритму, а вихід мережі встановлюється в необхідне значення 77.

Для демонстрації роботи генетичного алгоритму були підібрані наступні значення вхідних параметрів *in1*, *in2*, *in3*, *Zout*, а також значень початкової популяції, що весь процес налаштування вагових коефіцієнтів нейромережі

завершився за 3 ітерації, що і представлено на Рис. 3.13. У середовищі моделювання ISE Simulator (ISim) крок в 1 пікосекунду на кожну операцію генетичного алгоритму заданий програмно для налагодження та демонстрації роботи. Час навчання склав 0,15 мілісекунд.

Також були промодельовані процеси налаштування вагових коефіцієнтів нейромереж з 2 і 3 нейронами. Перша нейромережа складається з двох послідовно з'єднаних нейронів та 4-х синапсів. Час навчання склав 0,2 мілісекунди.

Друга ШНМ складається з двох нейронів в першому, вхідному, шарі і одного нейрона в другому, вихідному, шарі. Нейрони об'єднані чотирма синапсами. Час навчання склав 0,23 мілісекунди.

Порівняння отриманих результатів з еквівалентними результатами, виявленими у аналогічних опублікованих роботах[109, 110, 112, 113], представлені наступним чином. Порівняння, які узагальнені в таблиця 3.3, були зроблені з якомога більшою подібністю параметрів та реалізовані на однаковому чіпі. У таблиці представлений стовпець із порівняльними посиланнями, наступні два стовпці показують параметри генетичного алгоритму в відповідних посиланнях, N – розмір хромосоми, K – кількість епох генетичного алгоритму. В наступних стовпчиках показані часи, отримані роботами на аналогічному чіпі, результати отримані тут. також відображаються відповідні прискорення.

Таблиця 3.3 – Порівняння отриманих результатів з іншими результатами

Посилання	N	K	T_1	T_2	Прискорення
[112]	20	380	74 мс	0,285 мс	260
[113]	32	200	1,6 мс	0,2 мс	8
[109]	64	500	0,941 мс	0,625 мс	1,5
[110]	16	256	0,8 мс	0,23 мс	3,5

З результатів моделювання можна зробити висновок, що розроблений метод навчання нейронних мереж генетичним алгоритмом при апаратній реалізації на FPGA дозволяє значно збільшити швидкість адаптації нейромережевих компонентів систем керування чим підвищить їх ефективність.

3.4 Розробка алгоритму апаратної реалізації фільтра Калмана

Для фільтрації даних, що надходять на вхід нейронної мережі потрібно реалізувати фільтр. В якості фільтра був обраний фільтр Калмана, як оптимальний з точки зору складності реалізації, а отже і швидкодії та займаного ресурсу чіпа FPGA. Фільтр Калмана широко використовується в інженерних та економетричних додатках: від радарів та систем технічного зору до оцінок параметрів макроекономічних моделей. Фільтр Калмана призначений для рекурсивного дооцінювання вектора стану апріорно відомої динамічної системи, тобто для розрахунку поточного стану системи необхідно знати поточне вимірювання, а також попередній стан самого фільтра. Таким чином, фільтр Калмана, подібно до інших рекурсивним фільтрам, реалізований в тимчасовому, а не в частотному поданні, але на відміну від інших подібних фільтрів, фільтр Калмана оперує не тільки оцінками стану, а ще й оцінками невизначеності (щільності розподілу) вектора стану, спираючись на формулу Байеса умовної ймовірності.

Алгоритм працює в два етапи. На етапі прогнозування фільтр Калмана екстраполюють значення змінних стану, а також їх невизначеності. На другому етапі за даними вимірювання (отриманого з деякою погрішністю) результат екстраполяції уточнюється. Завдяки покрокової природі алгоритму, він може в реальному часі відстежувати стан об'єкта (без заглядання вперед, використовуючи тільки поточні виміри і інформацію про попередньому стані і його невизначеності).

Апаратна реалізація фільтра Калмана на FPGA дозволить синтезувати велику кількість паралельно працюючих фільтрів на одному кристалі, що в

свою чергу дає можливість синтезу систем керування багатовимірними об'єктами(MIMO) та систем здатних одночасно зчитувати та опрацьовувати дані з великої кількості датчиків та пристроїв. Апаратна реалізація фільтра Калмана на FPGA, виконується за наступним алгоритмом:

Крок 1: Виконати ініціалізацію змінних.

Задати дисперсію похибки датчика *sigmaeta*, середнє значення квадрату похибки на *k*-й ітерації *eopt_prev*, дисперсію похибки моделі *sigmaksi*. Задати *a,* коефіцієнт деякої відомої керуючої функції $u_k=a\cdot k$, де k – номер ітерації. При обчисленні використовувались десяткові числа, проте, так як в мові VHDL немає вбудованих засобів представлення не цілих десяткових чисел, то в програмному коді фільтра застосовуються коефіцієнти-змінні h=10000 і l=10 для перенесення коми, множення не цілих змінних в формулах. Таким чином отримуються великі цілі числа типу integer, з якими можна за допомогою вбудованих арифметичних операцій виконати необхідні обчислення. Отже в обчисленнях використовуються змінні з точністю до 0.0001. Результат представлений тільки цілою частиною.

Крок 2: Якщо на синхровхід надходить імпульс(перехід з 0 в 1) – перейти до **Крок 3**, інакше – чекати, поки не надійде імпульс (повторити **Крок 2**).

Крок 3: Зчитати з входу сигнал FILTER_IN покази датчика *z*<=FILTER_IN. *z* – покази датчика.

Крок 4: Обчислити квадрат середнього значення квадрату похибки

$$eopt:=divide(\ (sigmaeta^2)\cdot(eopt\_prev+sigmaksi^2),$$
$$(sigmaeta^2+eopt\_prev+sigmaksi^2)\ )$$

Вбудованої операції ділення двох десяткових чисел в VHDL не передбачено, тому операцію ділення (ф-я *divide*) в вищенаведеній формулі було реалізовано відновлювальним алгоритмом ділення. Ця функція приймає два числа типу integer (ділене та дільник) і приводить їх до типу unsigned. Тому отримавши два двійкових числа, дуже легко знайти частку маніпулюючи бітовими зсувами, ця операція швидка й використовує манімальну кількість ресурсів.

Крок 5: Обчислити коефіцієнти підсилення Калмана

$$k:=divide((eopt \cdot h),(sigmaeta^2));$$

В цій формулі також використовується перенесення коми і функція ділення *divide*.

Крок 6: Обчислити оптимальне відфільтроване значення

$$xopt:=(divide(xopt\_prev \cdot l,\ h)+a \cdot t \cdot l) \cdot (1 \cdot h\text{-}k)/l+k \cdot z;$$

де *z*, як було вказано вище, покази датчика.

Крок 7: Ділимо на коефіцієнт *h*

$$xoptout:=xopt/h;$$

і результат фільтра (*xoptout*). Дрібна частина з точністю до 0.0001 враховується в обчисленнях, але в результаті нехтується.

Крок 8: Видати результат на вихід фільтра *FILTER_OUT*:

FILTER_OUT <= xoptout

Крок 9: Інкремент кроку ітерації *k*=*k*+1.

Розглянемо приклад реалізації фільтра Калмана на FPGA. Для моделювання його роботи побудуємо схему в програмному забезпеченні Xilinx ISE Design Suite 13.2, на якій в свою чергу змоделюємо роботу ОК, датчика та фільтра Рис. 3.14.

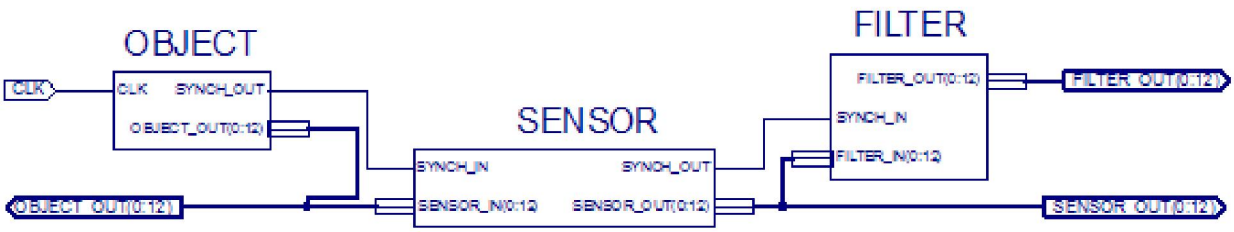


Рис. 3.14 – Схема моделі для дослідження роботи фільтра Калмана в САПР Xilinx ISE Design Suite 13.2

На Рис. 3.14 блок Object – модель об'єкта. Вхід: CLK – вхід синхронізації. Виходи: SYNCH_OUT – вихід синхронізації; OBJECT_OUT – вихідна величина об'єкта. Блок SENSOR – модель датчика. Входи: SYNCH_IN – вхід синхронізації (з об'єктом); SENSOR_IN – вхід, на який надходить вимірювана

величина. Виходи: SYNCH_OUT – вихід синхронізації(з фільтром); SENSOR_OUT – вихід, «виміряна» величина. Блок FILTER – фільтр Калмана. Отримує результати вимірювання датчика та за методом Калмана генерує значення, які ближчі до реальної вимірюваної величини**.** Входи: SYNCH_IN – синхронізація; FILTER_IN – вхід фільтра (для датчика з похибкою). Виходи: FILTER_OUT– оптимальне згенероване значення вимірюваної величини.

Нехай динаміка об'єкта керування описується фізичним законом руху. Характеризується деякою вихідною величиною, яка змінюється за заданим законом $x_{k+1} = x_k + v_k dt$, де $k \in [0,100]$ – номер ітерації; $x_{k+1} \in [0,4950]$ – вихідна(вимірювана) величина на ітерації k+1; a=1 – коеф. нелінійного закону; $v_k = a \cdot k$ – відома керуюча функція. В побудованій моделі похибка датчика не перевищує $\eta_k = 50$, генерується псевдовипадкова послідовність значень.

На часовій діаграмі, що зображена на Рис. 3.15, видно, що сигнал на виході датчика (sensor_out) спотворений похибкою, видно сигнал без похибки та роботу фільтра Калмана. При точності 0.0001 використовуваній в обчисленнях та досить великому проміжку значень вимірюваної величини, точність на виході фільтра знаходиться в межах 0.0001.

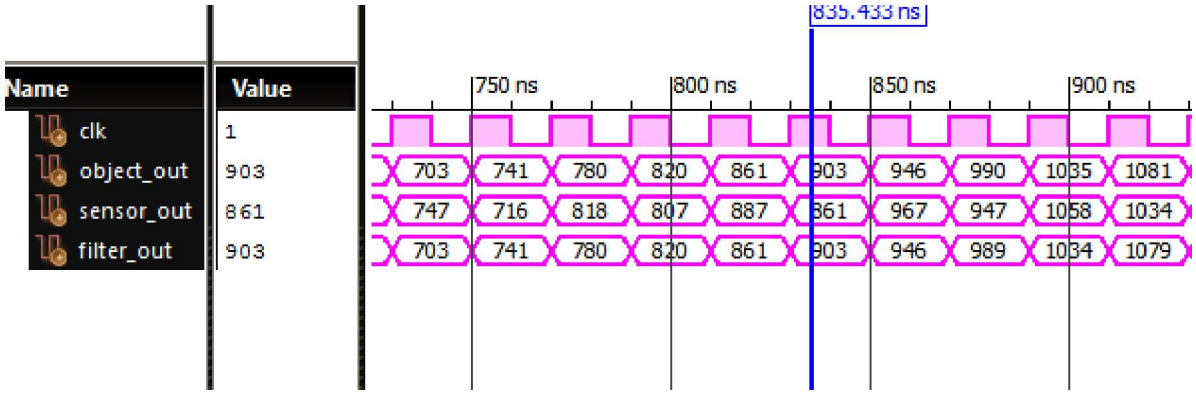


Рис. 3.15 – Моделювання роботи фільтра Калмана в програмному пакеті ISim

Наглядний графік фільтрації сигналу за методом Калмана датчика (sensor_out), відфільтрованого значення(filter_out) та реальної величини(object_out) побудований в Matlab, представлений на Рис. 3.16.

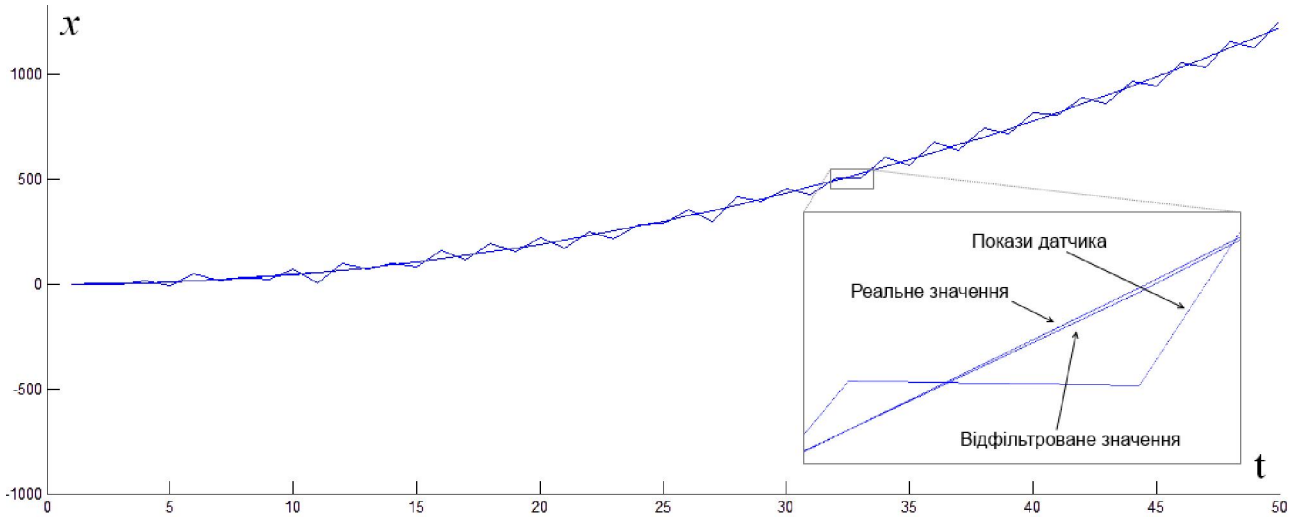


Рис. 3.16 – Графік фільтрації сигналу по Калману

Як видно з результатів дослідження, апаратний блок з фільтром Калмана дозволяє відфільтровувати завади на вхідних даних нейромережевих регуляторів систем керування.

3.5 Висновки до розділу 3

До основних результатів, отриманих в даному розділі, слід віднести наступне.

Дістав подальший розвиток комплексний формалізований підхід до реалізації багатоетапної процедури ідентифікації багатовимірних динамічних об'єктів з використанням нейронних мереж. Відрізняється від існуючих етапом створення сукупності нейромережевих моделей та їх оцінювання. Результатом отриманим за допомогою цієї технології є сукупність нейромережевих моделей з оцінкою їх точності по параметрам вхід-вихід. Це дозволяє обрати мінімальну структуру нейромережевої моделі, що задовольняє умови точності відтворення об’єкта. Для подальшої апаратної реалізації та пришвидшення роботи та навчання таких моделей важливим параметром є мінімальна архітектура моделі.

Розроблено метод проєктування апаратних компонентів нейромережевих систем, таких як пряма та інверсна модель об'єкта керування, на програ-

мованих логічних інтегральних схемах, відрізняється від існуючих етапом генерації та дослідження нейромережевих моделей, їх оцінки, та етапом реалізації нейромережевої моделі на FPGA, що дозволяє підвищити рівень автоматизації проєктування нейромережевих моделей при їх апаратній реалізації;

Розроблено метод проєктування апаратного компоненту оптимізації вагових коефіцієнтів нейронних мереж за допомогою генетичного алгоритму при реалізації його на програмованих логічних інтегральних схемах, який відрізняється від існуючих реалізацією операцій мутації та кросовера, що дозволяє значно підвищити швидкість оптимізації вагових коефіцієнтів нейронних мереж.

Розроблено алгоритм реалізації фільтра Калмана, розглянуто приклад його реалізації та промодельовано його роботу. Апаратний блок з фільтром Калмана дозволяє відфільтровувати завади на вхідних даних нейромережевих регуляторів систем керування.

РОЗДІЛ 4. ДОСЛІДЖЕННЯ НЕЙРОМЕРЕЖИВИХ СИСТЕМ КЕРУВАННЯ ДИНАМІЧНИМИ ОБ'ЄКТАМИ ЗІ ЗАСТОСУВАННЯМ АПАРАТНИХ КОМПОНЕНТІВ НА FPGA

Для дослідження нейромережевих систем керування складними динамічними об'єктами на основі апаратних компонентів реалізованих на FPGA задамо багатовимірний об'єкт керування у вигляді (1.1):

$$\begin{aligned} \dot{x}_1 &= -2x_1 - 2.5x_2 + 4u_1 + u_2, \\ \dot{x}_2 &= 4x_1 + u_2, \\ y_1 &= 0.25x_1 + 6.25x_2 + u_1 + u_2, \\ y_2 &= 0.1x_1 + x_2 + 1.25u_1 + u_2, \\ y_3 &= 2x_1 + 3x_2 + 0.5u_1 + u_2. \end{aligned} \tag{4.1}$$

Даний об'єкт має два входи u_1, u_2 та три виходи y_1, y_2, y_3, рисунок 4.1, описується матрицями **A**, **B**, **C**, **D**:

$$\mathbf{A} = \begin{pmatrix} -2 & -2.5 \\ 4 & 0 \end{pmatrix}; \quad \mathbf{B} = \begin{pmatrix} 4 & 1 \\ 0 & 1 \end{pmatrix}; \quad \mathbf{C} = \begin{pmatrix} 0.25 & 6.25 \\ 0.1 & 1 \\ 2 & 3 \end{pmatrix}; \quad \mathbf{D} = \begin{pmatrix} 1 & 1 \\ 1.25 & 1 \\ 0.5 & 1 \end{pmatrix}. \tag{4.2}$$

На Рис. 4.1 зображена структура об’єкта керування, що описаний матрицями (4.2), на Рис. 4.1 сигнали $u_1(t), u_2(t)$ – функції керування, $x_1(t), x_2(t)$ – стани об’єкта, $y_1(t), y_2(t), y_3(t)$ – функції виходу об’єкта

Рис. 4.1 – Структура об'єкта управляння

Для побудови моделі даного об'єкта в системі автоматичного проєктування Xilinx ISE Design Suite необхідно перейти до його цифрової форми. В цифровому вигляді, з частотою дискредитації T_0 = 0.01, (9.28) даний об'єкт буде описуватися такими матрицями:

$$\mathbf{A}=\begin{pmatrix}0.9797 & -0.0248\\ 0.0396 & 0.9995\end{pmatrix};\mathbf{B}=\begin{pmatrix}0.0396 & 0.0098\\ 0.0008 & 0.0102\end{pmatrix};\mathbf{C}=\begin{pmatrix}0.25 & 6.25\\ 0.1 & 1\\ 2 & 3\end{pmatrix};\mathbf{D}=\begin{pmatrix}1 & 1\\ 1.25 & 1\\ 0.5 & 1\end{pmatrix}. \quad (4.3)$$

На Рис. 4.2 зображено схему перевірочної моделі в системі MatLab, представлено неперервну модель ОК та дискретну модель ОК.

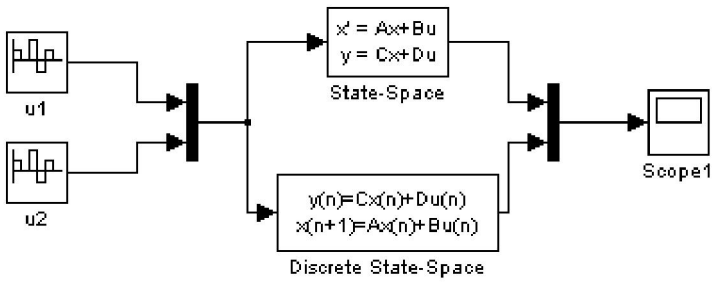


Рис. 4.2 – Модель неперервного та дискретного вигляду об'єкта керування

На Рис. 4.3 зображено графіки вхідних $u_1(t), u_2(t)$ та вихідних $y_1(t), y_2(t), y_3(t)$ функцій ОК.

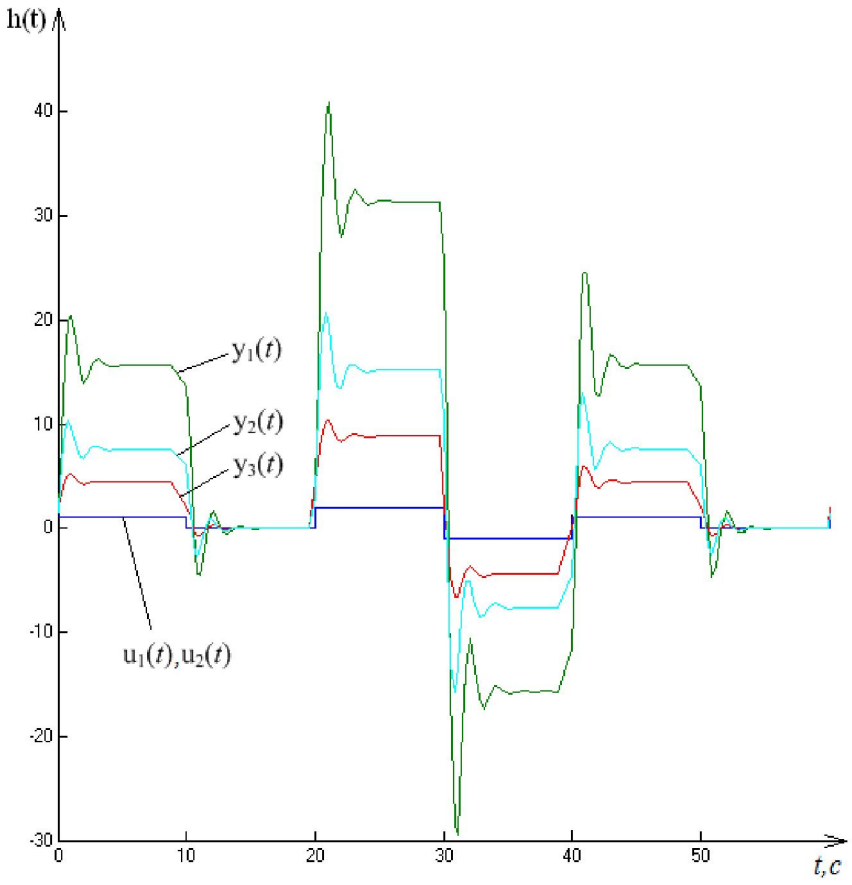


Рис. 4.3 – Перехідні характеристики об'єкта керування в неперервному та дискретному вигляді

Для синтезу нейромережевих моделей об'єкта керування використаємо метод синтезу апаратних компонентів систем керування на основі ШНМ на прикладі реалізації узагальненої нейромережевої моделі об'єкта керування, запропонований в 3.2 та проведемо синтез та дослідження систем керування без зворотнього звязку, системи керування з зворотнім зв’язком, системи керування з прямою та оборенною ШНМ моделю об’єкта та зворотнім зв’язком, системи керування з еталонною моделлю та зворотнім зв’язком.

4.1 Дослідження системи керування без зворотного зв'язку

Для синтезу та дослідження системи керування без зворотного зв'язку реалізуємо нейроконтролер та нейроемулятор представлені на Рис. 1.8. За розробленим методом описаним в підрозділі 3.2 реалізуємо нейромережеві компоненти такі як, пряма та обернена(інверсна) модель об'єкта керування. Компонент схеми, що відображає сам об'єкт керування реалізується за рівняннями простору станів в дискретному вигляді за матрицями (4.2). На Рис. 4.4 представлена модель системи керування без зворотного зв'язку з НК та НЕ схемному редакторі Xilinx ISE Design Suite 13.2.

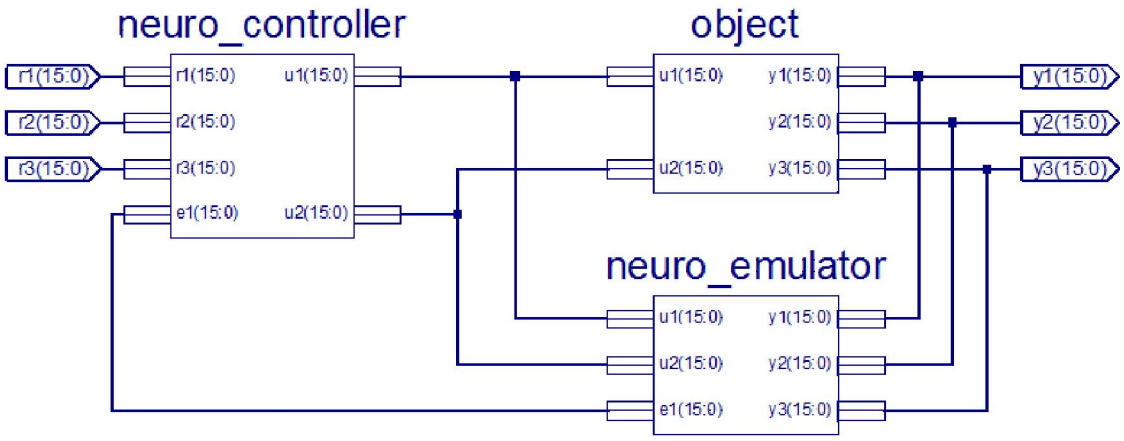


Рис. 4.4 – Модель система керування без зворотного зв'язку з НК та НЕ в схемному редакторі Xilinx ISE Design Suite 13.2

На Рис. 4.4 представлені *neuro_controller*_1 – обернена(інверсна) нейромережева модель об'єкта керування та компонент її адаптації. Адаптація проходить в режимі «on-line» за методом описаним в підрозділі 3.3. *Neuro_emylator* – пряма нейромережева модель об'єкта керування побудована в режимі «off-line». На *neuro_controller*_1 надходить сигнал *e*1 який є похибкою – різницею між виходом реального об'єкта та виходом моделі об'єкта.

На Рис. 4.5 представлено перехідний процес в системі керування без зворотного зв'язку з НК та НЕ.

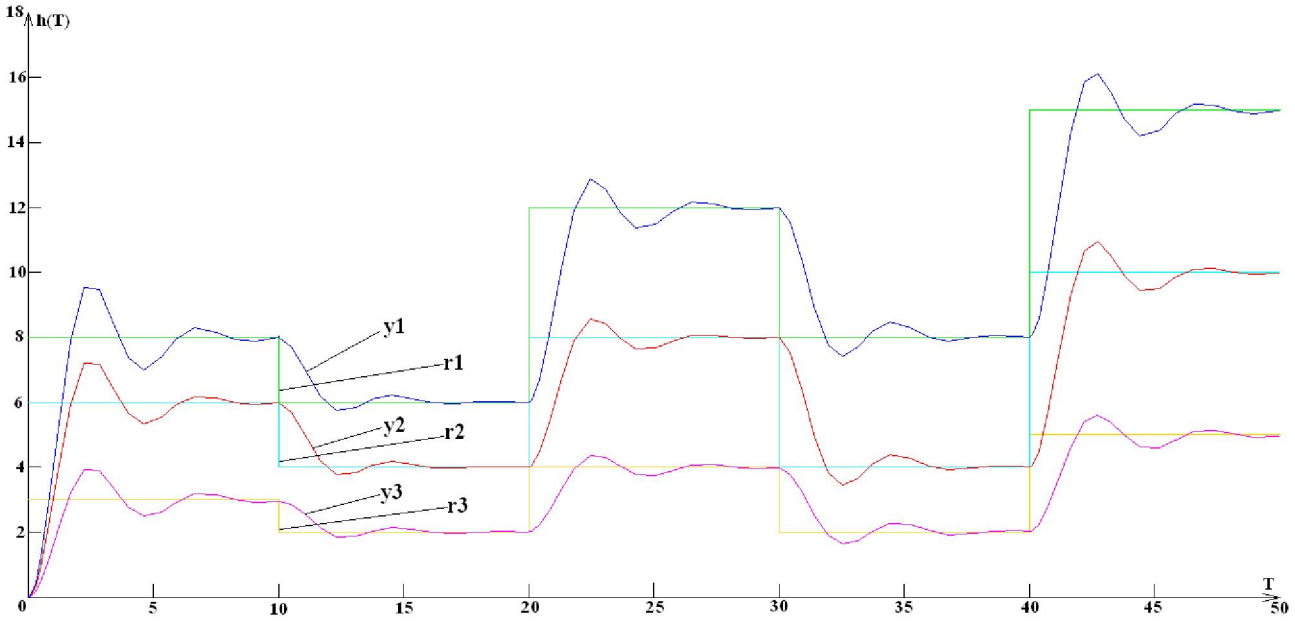


Рис. 4.5. – Перехідний процес в СУ без зворотного зв’язку з НК та НЕ

На Рис. 4.5 по осі ординат зображено задане значення, а по осі абсцис такти. Перегулювання в даній системі склало 20%, а час встановлення 7 тактів.

4.2 Дослідження системи керування зі зворотнім зв'язком

Для реалізації та дослідження системи керування з зворотнім зв'язком побудуємо схему системи керування з нейроконтролером представлену на

рисунках 1.6б та 1.9. Схема є класичною схемою спеціалізованого інверсного навчання. За розробленим методом реалізуємо нейромережевий компонент такий як, обернена (інверсна) модель об'єкта керування. На Рис. 4.6 представлена модель системи керування з зворотнім зв'язком в схемному редакторі Xilinx ISE Design Suite 13.2

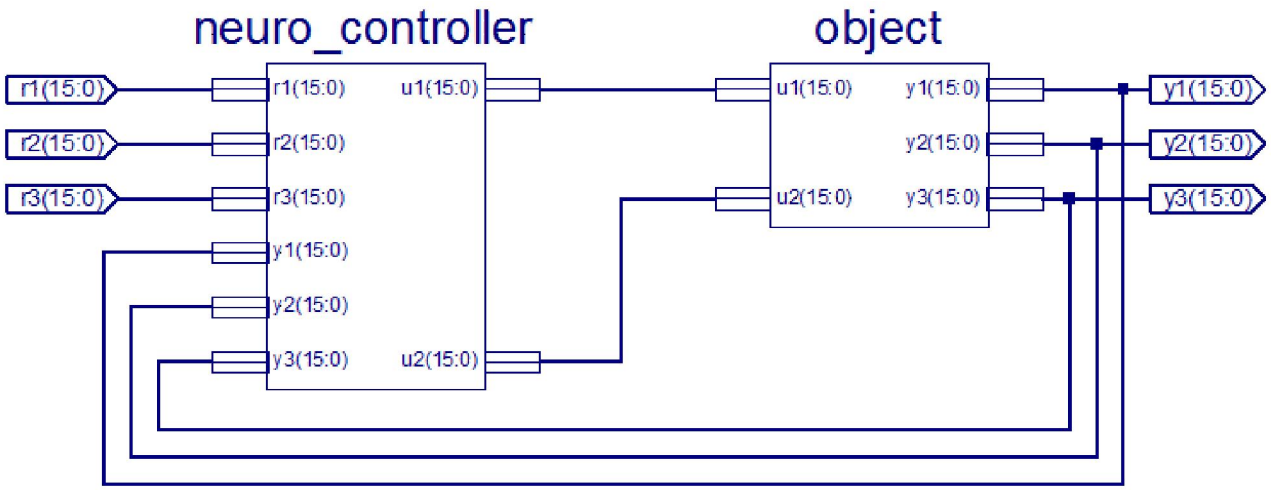


Рис. 4.6 – Модель системи керування з зворотнім зв'язком в схемному редакторі Xilinx ISE Design Suite 13.2

На Рис. 4.6 сигнали *neuro_controller*_2 – обернена(інверсна) нейромережева модель об'єкта керування та компонент її адаптації. Адаптація проходить в режимі «on-line» за методом описаним в підрозділі 3.3. На *neuro_controller*_1 надходять сигнали $y1$, $y2$ та $y3$ що є виходом об'єкта керування. Критерієм адаптації(фітнес-функцією генетичного алгоритму) є різниця між заданим значенням та вихідним значенням об'єкта керування.

На Рис. 4.7 представлено перехідний процес в системі керування з зворотнім зв'язком та НК в вигляді оберненої(інверсної) моделі об'єкта керування, по осі ординат зображено задане значення, а по осі абсцис такти. Перегулювання в даній системі склало 15%, а час встановлення 6 тактів.

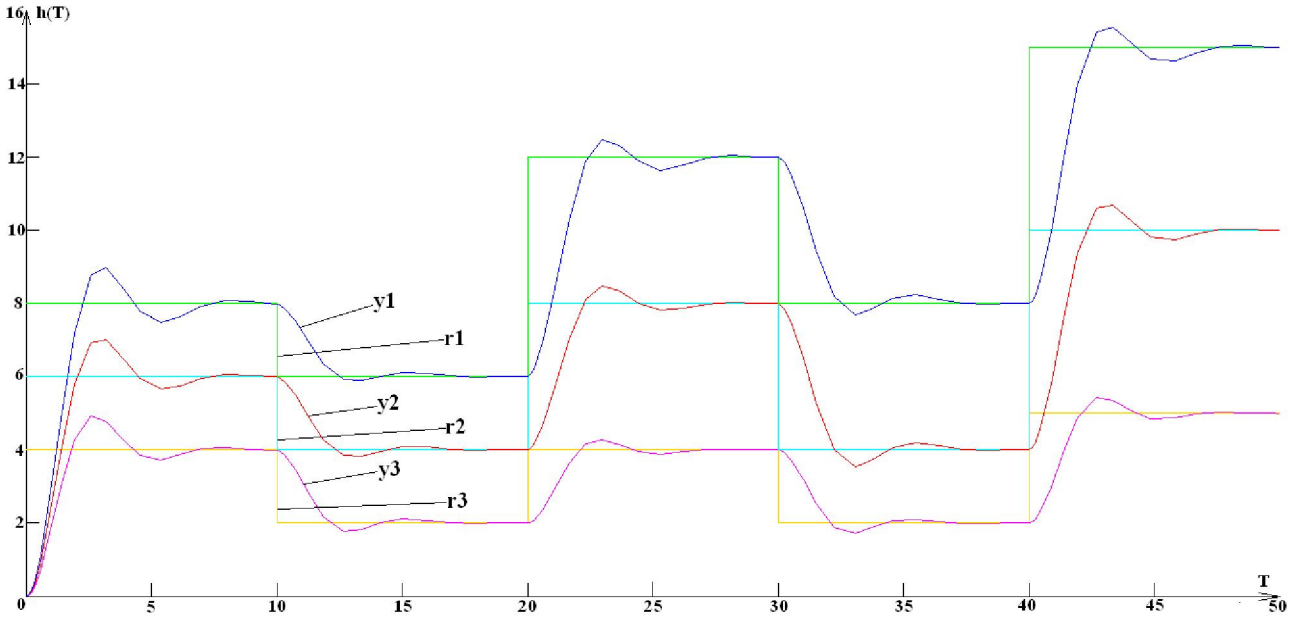


Рис. 4.7. – Перехідний процес в СУ зворотнім зв’язком

Для реалізації та дослідження адаптивної системи керування із прямою і інверсною моделями об'єкта керування побудуємо схему, що представлена на рисунку 1.16. За методом розробленим в підрозділі 3.2 синтезуємо нейромережеві компоненти схеми такі як, обернена та пряма модель об'єкта керування, також синтезуємо алгоритми їх адаптації за методом розробленим в підрозділі 3.3 та фільтри Калмана за алгоритмом розробленим в 3.4.

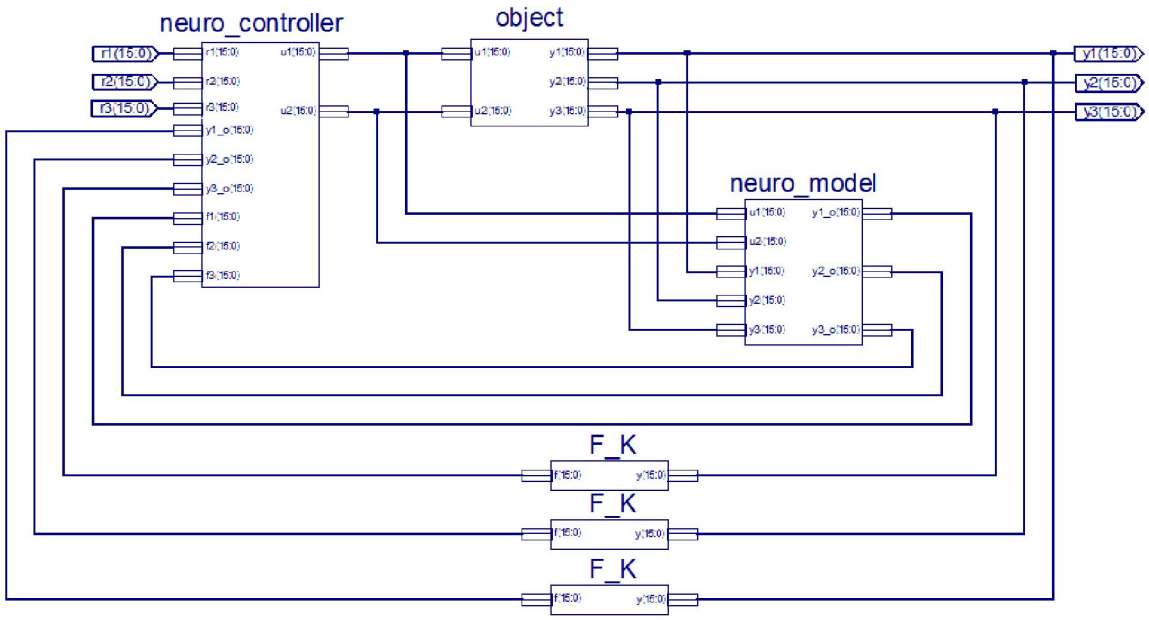


Рис. 4.8 – Модель адаптивної СУ із прямою та інверсною моделями об'єкта керування в схемному редакторі Xilinx ISE Design Suite 13.2

На Рис. 4.8 представлені *neuro_controller*_3 – обернена(інверсна) нейромережева модель об'єкта керування та компонент її адаптації, *neuro_model* – пряма нейромережева модель об'єкта керування. Адаптація проходить в режимі «on-line». На *neuro_controller*_3 надходять сигнали *y*1, *y*2 та *y*3 що є виходом об’єкта керування. Критерієм адаптації(фітнес-функцією генетичного алгоритму) є різниця між заданим значенням та вихідним значенням об'єкта керування.

На Рис. 4.9 представлено перехідний процес в системі керування з зворотнім зв'язком та НК в вигляді оберненої(інверсної) моделі об'єкта керування, по осі ординат зображено задане значення, а по осі абсцис такти. В даній системі час встановлення становить 6 тактів, перерегулювання 10%.

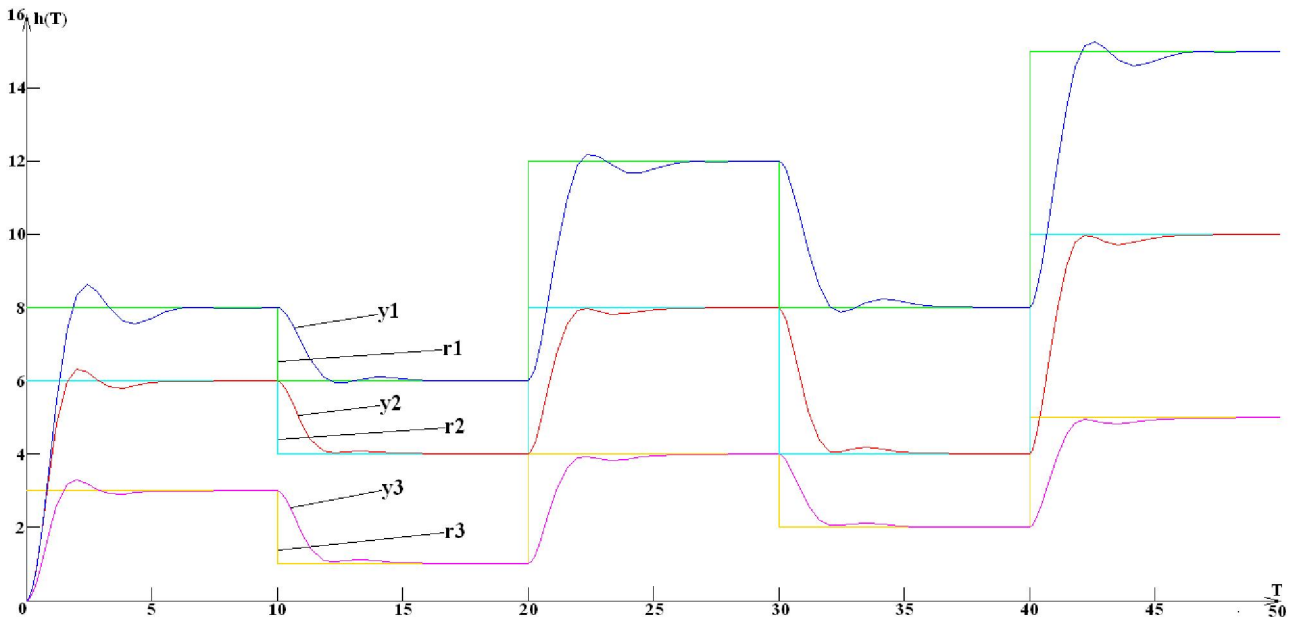


Рис. 4.9 – Перехідний процес в адаптивній СУ із прямою та інверсною моделями об'єкта керування

Для реалізації та дослідження адаптивної нейромережевої системи керування з еталонною моделлю побудуємо схему системи керування представлену на рисунку 1.17. За методом описаним в підрозділі 3.2 синтезуємо ней-

ромережевий компонент такий як, обернена(інверсна) модель об'єкта керування. На Рис. 4.10 представлена модель системи керування з зворотнім зв'язком в схемному редакторі Xilinx ISE Design Suite 13.2

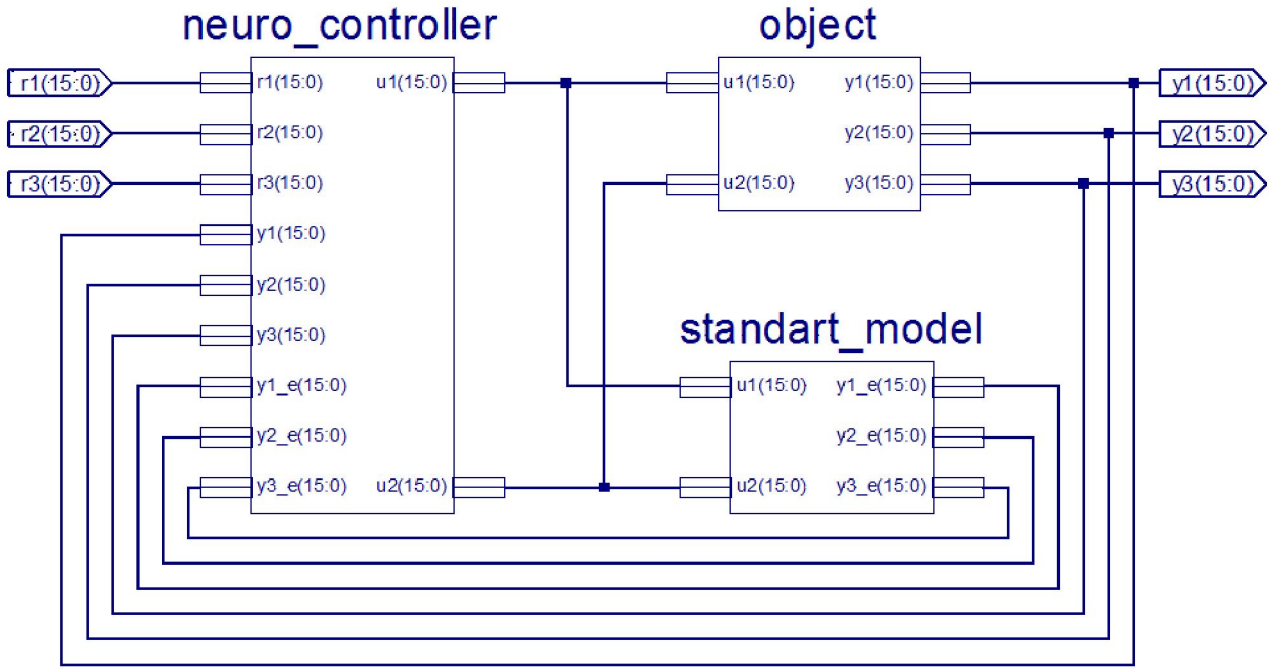


Рис. 4.10 – Модель системи керування з зворотнім зв'язком в схемному редакторі Xilinx ISE Design Suite 13.2

На рисунку 4.10 представлені *neuro_controller*_4 – обернена(інверсна) нейромережева модель об'єкта керування та компонент її адаптації. Адаптація проходить в режимі «on-line» за методом описаним в підрозділі 3.3. На *neuro_controller*_4 надходять сигнали *y*1, *y*2 та *y*3 що є виходом об'єкта керування та сигнали *y*1_e, *y*2_e та *y*3_e, що є виходами еталонної мережі. Критерієм адаптації(фітнес-функцією генетичного алгоритму) є різниця між заданим значенням та вихідним значенням об'єкта керування, різниця між виходом еталонної моделі та об'єкта керування.

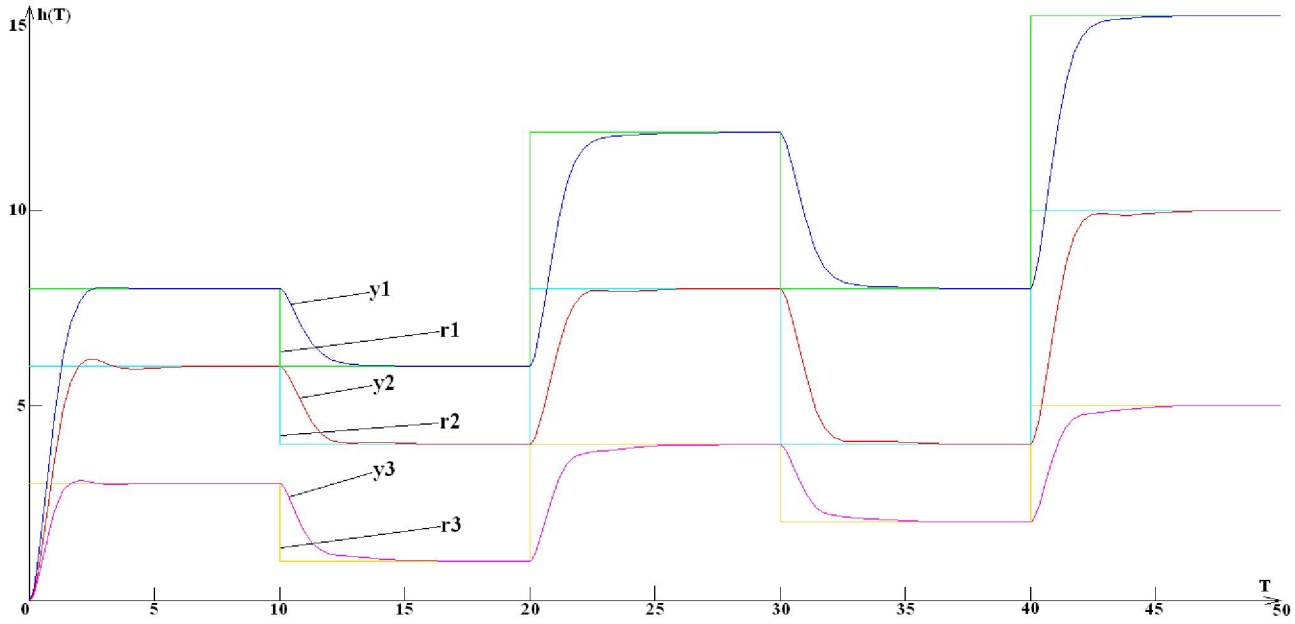


Рис. 4.11 – Перехідний процес в адаптивній нейромережевій СК з еталонною моделлю

На рисунку 4.11 представлено перехідний процес в адаптивній та оптимальній нейромережевій СК з зворотнім зв'язком, НК в вигляді оберненої(інверсної) моделі об'єкта керування та еталонної моделі об'єкта, по осі ординат зображено задане значення, а по осі абсцис такти. В даній системі час встановлення становить 3 такти.

4.3 Узагальнена структурна схема нейроконтролера

В даний час для вирішення ресурсномістких задач все частіше застосовується апаратні прискорювачі, у тому числі і реконфігуровані уніфіковані обчислювачі.

Головна проблема яка стримує застосування реконфігурованих уніфікованих обчислювачів – складність створення конфігурацій для FPGA. Під конфігурацією будемо розуміти набір конфігураційних бітів (bitstream) у вигляді файлу даних, що завантажується в FPGA та формує її внутрішню структуру. Процес створення конфігурацій для мікросхем FPGA є трудомісткою

задачею, оскільки вимагає від розробника високої кваліфікації, знань в області синтезу цифрових схем, володіння спеціалізованими пакетами.

Одним з варіантів вирішення задачі проєктування нейромережевих контролерів є розробка узагальненої структурної схеми на базі рекунфігурованих процесорів, що дозволяє задіяти однаковість типових рішень для реалізації відносно широкого класу додатків в заданій предметній області. Така узагальнена структурна схема дозволить підвищити швидкість розробки конфігурацій.

Сучасні засоби керування технічними та технологічними об'єктами мають в своєму складі обчислювальне ядро, засоби вводу аналогової та цифрової інформації, засоби виводу аналогової та цифрової інформації, засоби обміну даними з іншими обчислювальними пристроями, елементи візуалізації та оперативного керування. Такі засоби керування будуються на основі промислових комп'ютерів, вільно програмованих контролерів, спеціалізованих контролерів та мікроЕОМ. Далі запропонуємо узагальнену структуру контролера для керування динамічними об'єктами. В якому в якості обчислювального ядра FPGA.

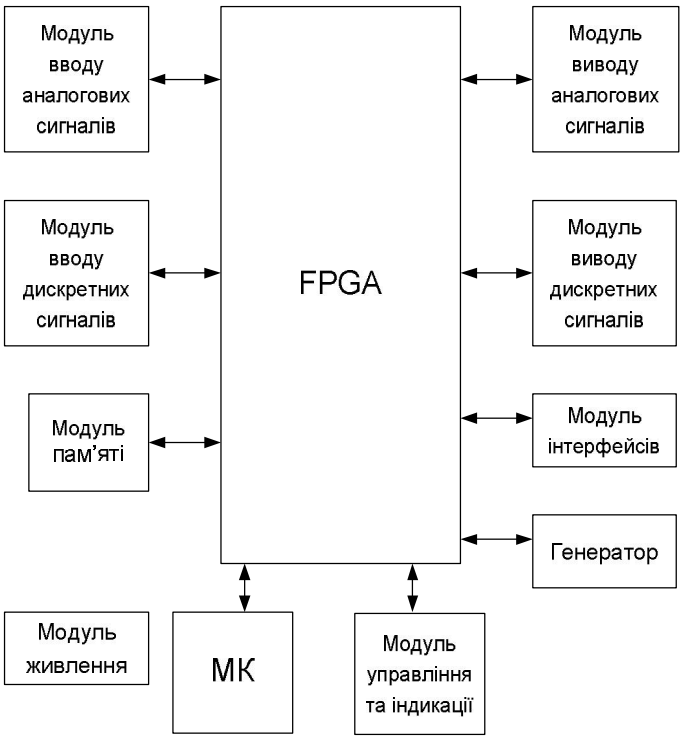


Рис. 4.12 – Узагальнена структурна схема нейроконтролера

На Рис. 4.12 представлена узагальнена структурна схема нейроконтролера, де в якості обчислювального ядра використовується нейрообчислювач на основі FPGA, що реалізує штучні нейроні мережі.

До складу нейроконтролера входять: FPGA – програмована логічна інтегральна схема; МК – мікроконтролер; модуль вводу аналогових сигналів; модуль виводу аналогових сигналів; модуль вводу дискретних сигналів; модуль виводу дискретних сигналів; модуль пам'яті; модуль інтерфейсів; модуль керування та індикації; генератор.

FPGA є центральним елементом даної структури, яка реалізує нейромережеві алгоритми керування, включаючи алгоритм навчання мережі в режимі «on line».

Мікроконтролер (МК) виконує допоміжні функції, такі як завантаження конфігураційної послідовності в мікросхему FPGA, реалізує функцію спостереження.

Модулі вводу/виводу аналогових та цифрових сигналів для вводу/виводу інформації з/в систему або об'єкт керування.

Модуль пам'яті забезпечує зберігання даних та конфігураційної послідовності FPGA.

В даній структурній схемі присутні два обчислювальних ядра: мікроконтролер (МК) і FPGA. В залежності від задач керування, мікроконтролер може бути: основним елементом нейромережевого контролера, а FPGA відводиться роль «швидкого» обчислювача фрагментів загального алгоритму роботи контролера пов'язаного з нейрообчисленями; доповнювати на рівних програму роботи FPGA, або бути чисто технічним елементом нейромережевого контролера, що забезпечує режими роботи FPGA. Але, в усіх випадках FPGA призначена для реалізації нейромережевих елементів системи керування, тому в подальшому основну увагу будемо приділяти обчислювачу на FPGA.

4.4 Нейромережевий контролер системи стабілізації рухомого об'єкта

Для побудови макета нейромережевого контролера в якості об'єкта керування обрано систему балансування кульки на платформі, здатну встановити кульку в задану точку. Платформа нахиляючись по кожній з двох горизонтальних осей контролює положення кульки. По кожній осі для нахилу платформи використано електричні двигуни. Положення кульки на платформі фіксується за допомогою відеокамери. Таке завдання балансування є розширенням задачі балансування «перевернутого маятника». Такий об'єкт керування є складним багатовимірним об'єктом та розроблений для демонстрації НСК синтезованих на розроблених компонентах.

Метою даної роботи є розробка автоматичної системи стабілізації рухомого об'єкта на площині в реальному часі з можливістю встановлення рухомого об'єкту в заданій точці на площині, використовуючи розроблені методи та алгоритми проєктування апаратних компонентів НСК.

Для побудови системи балансування кульки на платформі було вирішено декілька завдань. Першочерговим завданням є визначення положення кульки, точно, надійно і в не громіздкий та недорогий спосіб. На підставі розглянутих переваг і недоліків, пов'язаних з кожним вибором, було прийнято рішення відстежувати положення кульки за допомогою цифрової відеокамери. Наступним завданням було спроєктувати механізм нахилу пластини. Пластина має нахилятися по двох своїх осях, щоб мати можливість збалансувати кульку. Для цієї конструкції, обрано варіант коли два приводи прикріплюються по двох кутах пластини, яка підтримується кульовим шарніром в центрі, в якості двох необхідних ступенів руху.

На Рис. 4.13 показаний загальний вигляд стенду по балансуванню кульки на платформі, включаючи механізм нахилу пластини на серводвигунах і зчитування положення кульки за допомогою відеокамери.

Рис. 4.13 – Загальний вигляд стенду балансування кульки на платформі

Система балансування кульки на платформі складається з пластикової пластини на якій буде знаходитись кулька, приводного механізму для нахилу пластини навколо двох осей, цифрової відеокамери, для відстеження положення кульки, апаратного та програмного забезпечення, що опрацьовує інформацію та керує системою в режимі реального часу.

Для подальшого фізичного моделювання системи зробимо наступні припущення використовуються в моделюванні вищеописаної фізичної системи:

1. Передбачається, що тертя-ковзання між кулею і пластиною досить високе, щоб запобігти кульку від ковзання на пластині. Це обмежує ступені свободи системи, а також робить рівняння руху простішим.

2. Обертання кулі навколо своєї вертикальної осі, як передбачається, буде незначним.

3. Тертям-коченням між кулькою та пластиною – знехтувати.

4. Передбачається, що буде невеликий рух пластини при рівноважній конфігурації. Це гарантує, що кути пластини будуть приблизно рівні кутам двигуна.

5. Пластина, як передбачається, має максимальну симетрію.

Фізична модель керованого об'єкта представлена на Рис. 4.14, де *X-Y-Z* є підставою рами. Платформа має два ступені свободи, а його орієнтація визначається двома кутами (q_1 і q_2), які представляють собою тіла обертання. Рамка x"-y"-z" є пластина, закріплена система відліку, в той час як x'-y'-z' є проміжними.

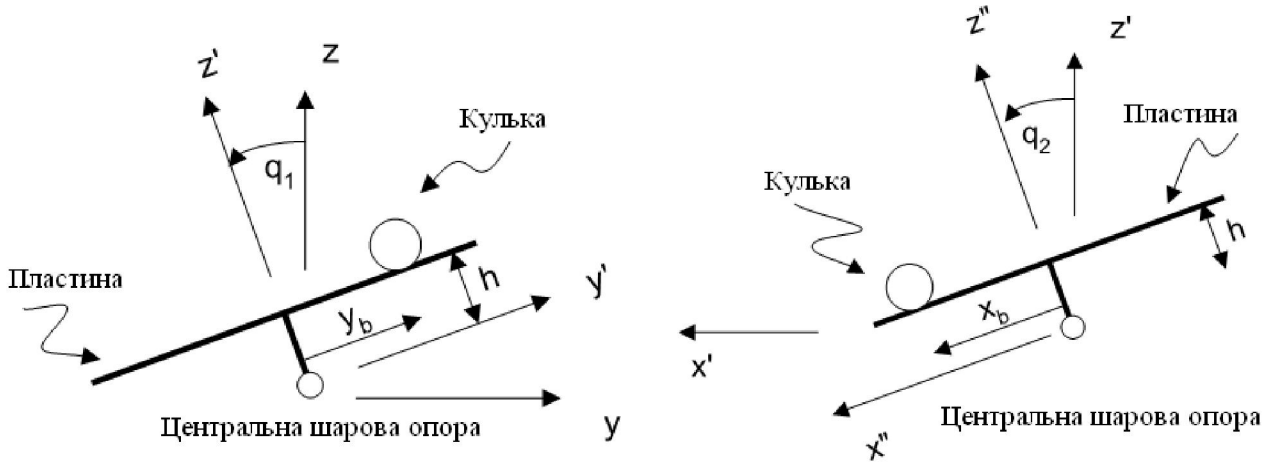


Рис. 4.14 – Фізичне моделювання руху кульки по платформі

Кількість надлишкових ступенів свободи два, а саме повороти двох вертикальних зв'язків по своїх осях складають два резервних ступеня свободи, так як вони не впливають на стан системи в будь-якому випадку.

Таким чином, плита і механізм просторового зв'язку мають два ступені свободи, як і очікувалося. Це також дорівнює числу входів в систему, керуючий сигнал двома двигунами.

Виходячи з вищевикладеного, очевидно, що з чотирьох змінних – $\theta_{m1}, \theta_{m2}, q_1, q_2$, тільки дві є незалежними, так як механізм має два ступені свободи. Таким чином, існують наступні два кінематичні рівняння зв'язку, які трансформують кут повороту валів двигунів в кут нахилу платформи

$$\left(r_1\cos q_1 - r_1\cos\theta_{m1}\right)^2 + \left(r_1\sin q_1 - r_1\sin\theta_{m1} + l_1\right)^2 = l_1^2,$$
$$\left(r_2\cos\theta_{m2} - r_2\sin q_2\cos q_1 + l_2\right)^2 + \left(r_2\cos q_2 - r_2\cos\theta_{m2}\right)^2 + r_2\sin q_2\sin q_1 = l_2^2.$$
(4.4)

Помічено, що, в загальному випадку кут повороту пластини q_2 пов'язаний з кутами повороту двигунів θ_{m1} та θ_{m2} складними нелінійними рівняннями, наведеними вище. Проте, для малих рухів на рівновагу кульки на платформі, можна зазначені вирази звести до наступних лінійних виразів:

$$q_1 = \theta_{m1},$$
$$q_2 \cong \theta_{m2}.$$

Справедливість цього припущення також перевірена експериментально. Встановлено, що для відповідного діапазону роботи, відповідність між кутом повороту двигуна і кутом нахилу є задовільною.

Рівняння руху для цієї системи можуть бути отримані з кінематичних рівнянь, які трансформують кут повороту валів двигунів в кут повороту платформи, за допомогою законів Ньютона або рівнянь Лагранжа. Для цього випадку ці методи були використані для перевірки остаточних результатів. Повна нелінійна модель системи визначається за формулами:

x – координата:

$$m_b g r_b \sin q_2\cos q_1 - m_b r_b\begin{bmatrix}\left(h+r\right)\ddot{q}_2 - y_b\ddot{q}_1\sin q_2 - x_b\dot{q}_2^2 - x_b\dot{q}_1^2\sin^2 q_2 + \\ +\left(h+r_b\right)\dot{q}_1^2\sin q_2\cos q_2 - 2\dot{y}_b\dot{q}_1\sin q_2 + \ddot{x}_b\end{bmatrix} -$$
$$-I_b\left(\left(\ddot{x}_b/r_b\right) + \ddot{q}_2\right) = 0$$

(4.5)

y – координата:

$$m_b g r_b\sin q_1 -$$
$$-m_b r_b\begin{bmatrix}x_b\left(\ddot{q}_1\sin q_2 + \dot{q}_2\dot{q}_1\cos q_2\right) - \left(h+r_b\right)\left(\ddot{q}_1\cos q_2 + \dot{q}_2\dot{q}_1\sin q_2\right) + \\ +\dot{q}_2\dot{q}_1\left(h+r_b\right)\sin q_2 - \dot{y}_b\dot{q}_1^2 + x_b\dot{q}_2\dot{q}_1\cos q_2 + 2\dot{x}_b\dot{q}_1\sin q_2 + \ddot{y}_b\end{bmatrix} - \quad (4.6)$$
$$-I_b\left(\ddot{y}_b/r_b + \ddot{q}_1\cos q_2\dot{q}_2\dot{q}_1\sin q_2\right) = 0$$

Рівняння руху є нелінійними та малопридатні для синтезу системи керування на базі лінійної теорії керування.

Схема просторового зв'язку показана на Рис. 4.15. З'єднання жорстко прикріплене до основи пластини і з'єднане з землею в точці *O*. Два двигуни з'єднані з тягами нерухомими з'єднаннями. Решта на шарових шарнірах.

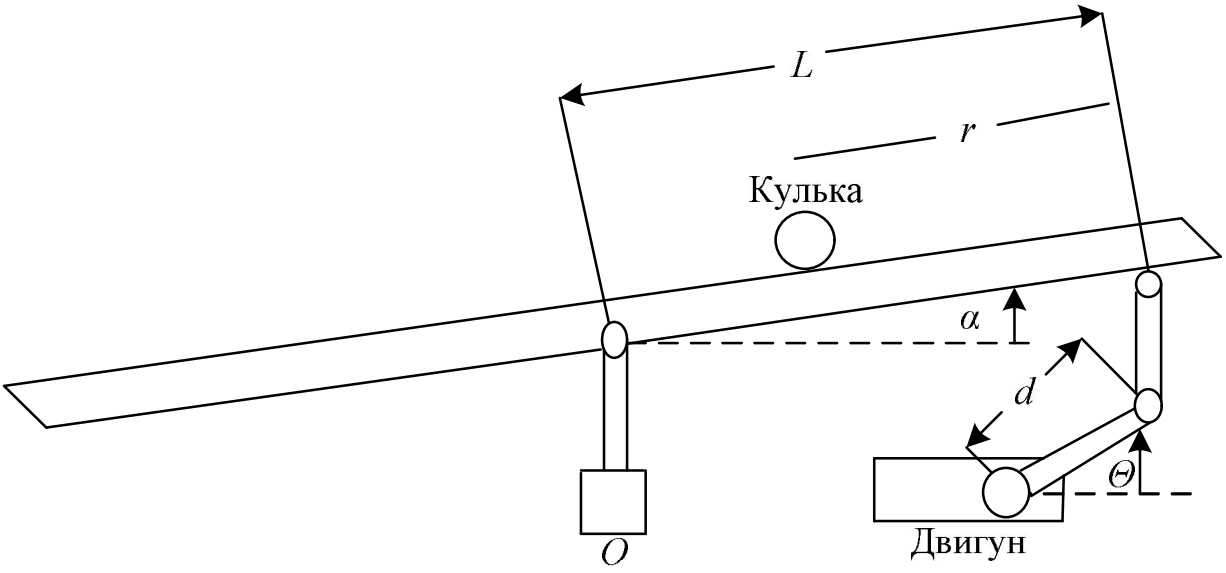


Рис. 4.15 – Механізм керування платформою

Для розрахунків та моделювання системи керування кулькою на платформі спростимо рівняння (4.5) та (4.6), припустимо, що ковзання і тертя при русі кульки відсутні. Розглянемо рівняння Лагранжа:

$$\left(\frac{J}{R^2}+m\right)\ddot{r}+mg\sin\alpha-mr\dot{\alpha}^2=0 \qquad (4.7)$$

де: R – радіус кульки (0.015м); m – маса кульки (0.01кг); d – довжина подовження серводвигуна (0.05м); g – прискорення вільного падіння; L – відстань від кінця до середини платформи (0.35м); J – момент інерції м'яча ($9.99\cdot10^{-6}$ кг·м^2); r – відстань від центру до краю платформи; α – кут повороту пластини; а Θ – кут повороту серводвигуна.

Якщо використовуються майже нульові кути положення платформи, можна припустити, що у випадку ряду Тейлора вони дорівнює куту $\sin\alpha$. Подібна ситуація для $\cos\alpha$, яка дорівнює одиниці, також можна знехтувати в градусах, починаючи з квадрата, наприклад, якщо швидкість становить 0.1

радіана в секунду, то квадрат швидкості дає дуже невеликий внесок, тому цим також можна знехтувати. Тому:

$$\alpha \approx 0 \rightarrow \begin{cases} \sin(\alpha) \approx \alpha \\ \cos(\alpha) \approx 1 \\ (\dot{\alpha})^2 \approx 0 \end{cases}$$

Спрощене рівняння (4.7) виглядає наступним чином:

$$\left(\frac{J}{R^2} + m\right)\ddot{r} = -mg\alpha \tag{4.8}$$

Кут повороту можна наблизити як відношення довжини серводвигуна і його кута повороту до відстані до середини платформи:

$$\alpha = \frac{d}{L}\theta \tag{4.9}$$

Підставимо рівняння (4.9) в рівняння (4.8) і отримуємо наступний вираз:

$$\left(\frac{J}{R^2} + m\right)\ddot{r} = -mg\frac{d}{L}\theta \tag{4.10}$$

Перейдемо до передавальної функції через перетворення Лапласа за допомогою рівняння (4.10):

$$\left(\frac{J}{R^2} + m\right)R(s)s^2 = -mg\frac{d}{L}\theta(s) \tag{4.11}$$

Виразимо передавальну функцію через відношення обертання двигуна до $\theta(s)$ положення кулі $R(s)$:

$$T(s) = \frac{R(s)}{\theta(s)} = -\frac{mg\dfrac{d}{L}}{\left(\dfrac{J}{R^2} + m\right)s^2}\left(\frac{m}{rad}\right) \tag{4.12}$$

З рівняння (4.12), підставляючи відповідні дані, ми отримуємо передавальну функцію:

$$T(s) = \frac{0.9482}{s^2}.$$

Отримана передавальна функція $T(s)$ промодельована в пакеті Matlab Simulink і отримана залежність положення кулі від кута повороту сервомотора, модель системи керування кулькою на платформі показана на рисунку 4.16.

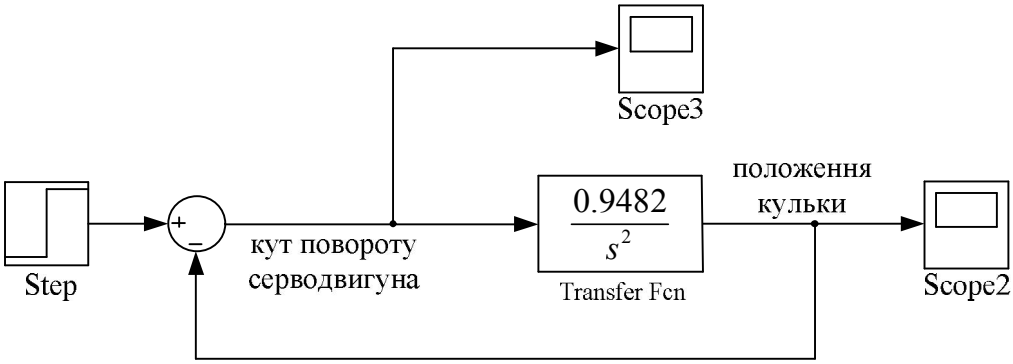
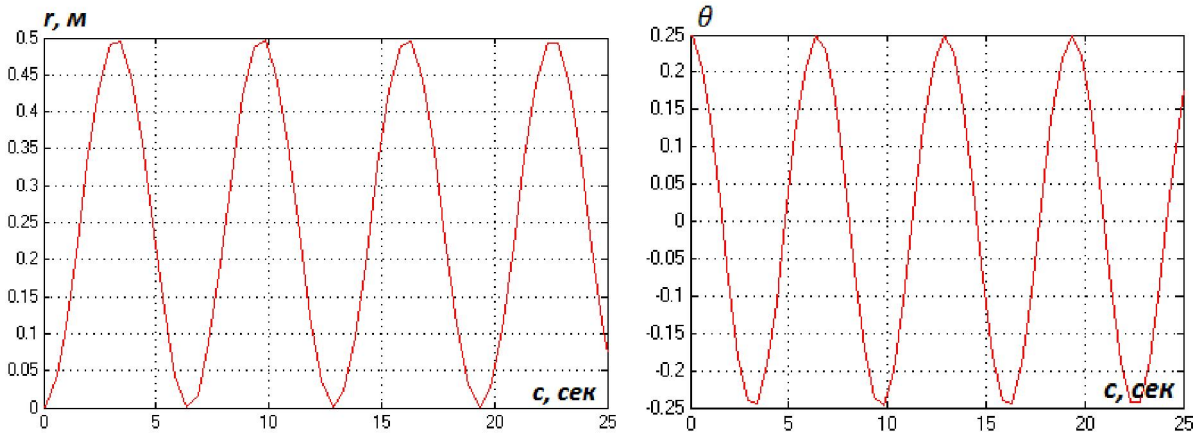


Рис. 4.16 – Модель системи курування рухом кулі на платформі в програмному пакеті Matlab Simulink

На рисунку 4.17(а) наведено графік положення кульки залежно від кута повороту серводвигуна, графік якого показаний на рисунку 4.17(б). Отже, коли двигун обертається на кут близько 0.17, кулька рухається з однієї сторони платформи до іншої.



(а) (б)

Рис. 4.17 – Моделювання керування рухом кулі на платформі в Matlab Simulink: (а) графік положення кулі, залежно від кута повороту серводвигуна; (б) графік кута повороту сервомотора.

Для керуванням рухом кульки, в залежності від кута повороту, була розроблена НСК з інверсною моделлю об'єкта керування та зворотнім зв'язком, що представлена на Рис. 4.18.

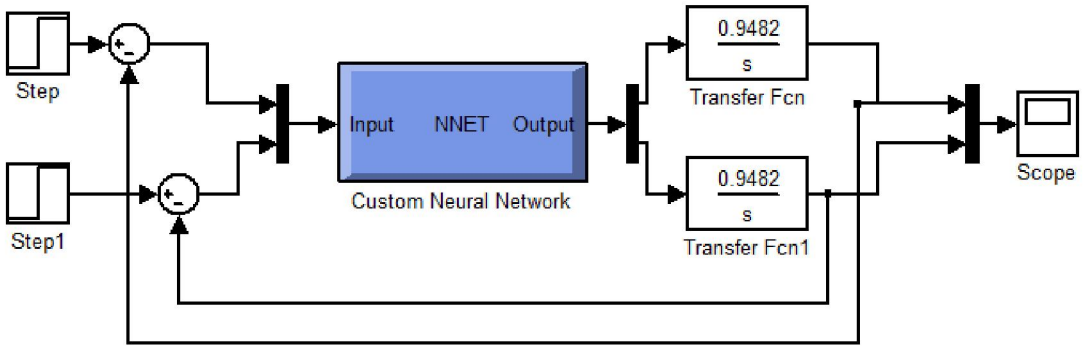
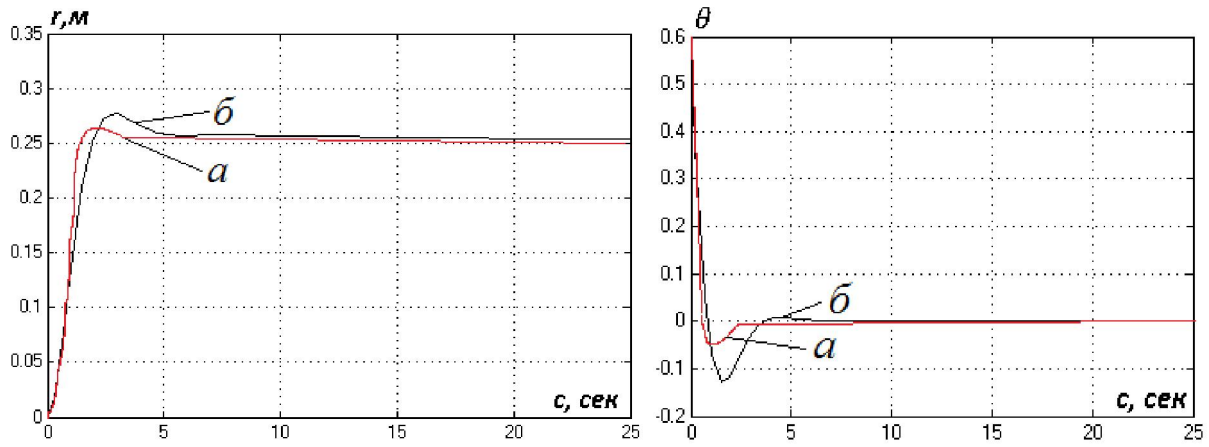


Рис. 4.18 – Модель НСК рухом кулі на платформі

Схема є класичною схемою спеціалізованого інверсного навчання. За допомогою технології і програмного пакету розроблених в підрозділі 3.1, була обрана тришарова ШНМ з двома нейронами в вхідному шарі, вісьмома нейронами в прихованому шарі та двома нейронами в вихідному шарі, дослідження різного типу нейромережевих моделей в розробленому середовищі «MIMO-plant» що дана топологія з найменьшою похибкою відтворить рух кульки на платформі. Початкове навчання інверсної нейромережевої моделі виконано на даних отриманих при моделюванні ПІД-регулятора в [123,138].

За методами описаними в третьому розділі синтезуємо нейромережевий компонент такий як, обернена(інверсна) модель об'єкта керування та компонент її адаптації. В процесі роботи стенду нейромережевий регулятор буде адаптуватися та нівелювати вісі ті припущення та невизначеності при моделюванні та розрахунках, а також до змін в умовах його роботи, наприклад при зміні параметрів кульки, до внесення збурюючи впливів в систему. На Рис. 4.19*а* та Рис. 4.19*б* криві *а* – перехідні процеси системи керування кулькою на платформі з НСК, криві *б* – перехідні процеси системи керування кулькою на платформі з ПІД-регуляторами[123, 138].



(*а*) (*б*)

Рис. 4.19 – Моделювання керування рухом кулі на платформі з НСК регулятором: (*а*) графік положення кулі, залежно від кута повороту серводвигуна, (*б*) графік кута повороту серводвигуна.

Апаратна частина стенду представлена на рисунку 4.20.

Рис. 4.20 – Структурна схема НСК рухом кулі на платформі

Робота стенда по керуванню рухом кулі на платформі за допомогою високошвидкісного, адаптивного, нейромережевого регулятора показано на рисунку 4.21.

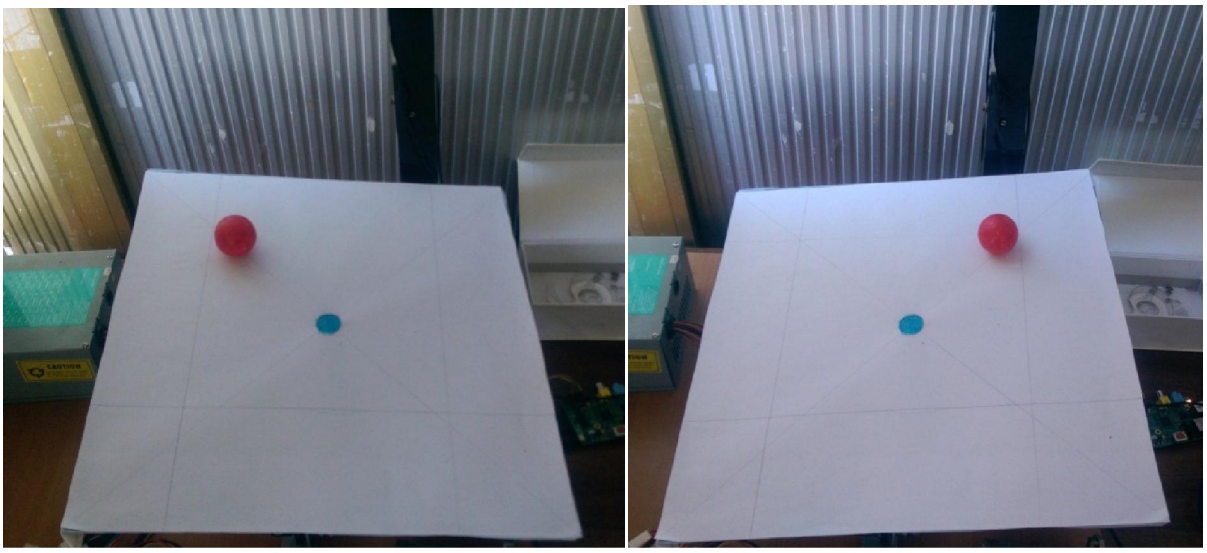


Рис. 4.21 – Процес керування кульки на платформі за допомогою контролера на FPGA

Для керування положенням кульки на платформі розроблено нейромережеву систему керування з зворотнім зв'язком, представлену на Рис. 4.22. Схема є класичною схемою спеціалізованого інверсного навчання. За методом описаним в третьому розділі реалізуємо нейромережевий компонент такий як, обернена(інверсна) модель об'єкта керування. На Рис. 4.22 нейроконтролер включає в себе інверсну модель об'єкта керування, компонент її налагодження, розроблений за методом описаним в третьому розділі та обернений зв'язок.

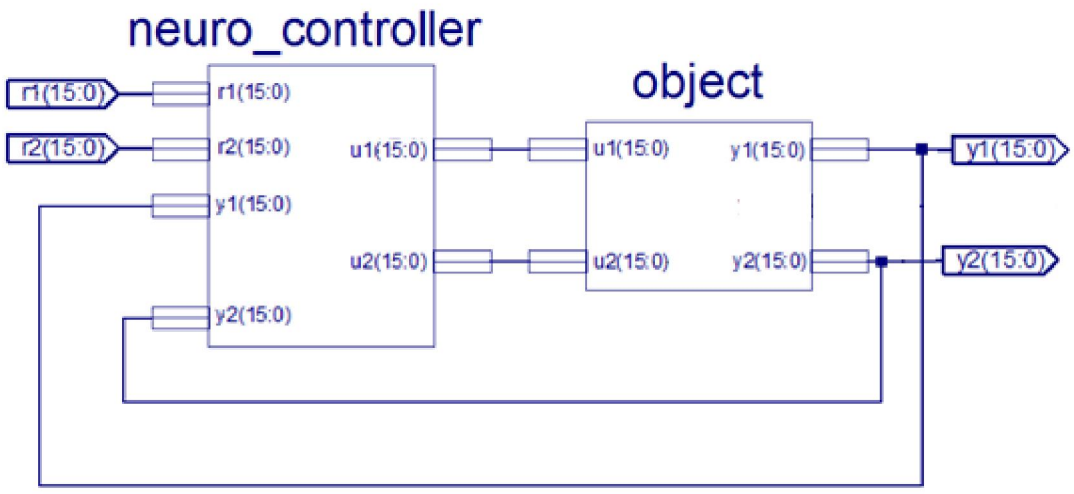


Рис. 4.22 – Модель системи керування

Схема представлена на Рис. 4.22 є моделлю всієї системи керування включно з об'єктом на основі розроблених раніше нейромережевих компонентах, в FPGA завантажено сам нейроконтролер.

4.5 Висновки до розділу 4

В даному розділі реалізовано та досліджено на основі розроблених компонентів: системи керування без зворотного зв'язку; системи керування з зворотнім зв'язком, адаптивна система керування із прямою та інверсною моделями об'єкта керування, адаптивної нейромережевої системи керування з еталонною моделлю, розроблено макет діючого нейроконтролера системи стабілізації рухомого об'єкта на обмеженій площині.

При дослідженні даних систем керування показано, що розроблені нейромережеві компоненти дозволяють будувати адаптивні системи керування складними багатовимірними об'єктами. Побудовані нейроконтролери, за розробленими методами, можуть функціонувати та адаптуватися в режимі реального часу та формувати функції керування будь-якої складності за рахунок використання оптимальних, за складністю, нейронних мереж, що реалізуються на FPGA.

Розроблено узагальнену структурну схему нейроконтролера на основі програмованих логічних інтегральних схем, що реалізує базові компоненти нейромережевих систем керування.

Реалізовано макет нейроконтроллера для системи стабілізації рухомого об'єкта на обмеженій площині та проведено його дослідження, які підтвердили високу ефективність роботи.

Такий об'єкт керування, як кулька на платформі є складнім нелінійним об'єктом тому для керування ним розроблено нейромережеву систему керування, яка відноситься до класу нелінійних динамічних систем. Розроблена в даній роботі нейромережева система керування функціонує та адаптуються в режимі реального часу та формує досить складні функції керування. Коли

система керування ввімкнена, кулька буде збалансована в будь-якій заданій точці пластини. Вона також може бути спрямована на перехід від однієї точки до іншої, і залишитися там. Використовуючи цю систему керування можна задати траєкторію руху кульки по платформі, наприклад рух по колу.

Побудовану і випробувану систему можна використовувати як відмінний випробувальний стенд для тестування різних інших схем керування. В подальшому можуть бути розроблені на базі FPGA більш оптимальні контролери для досягнення ще більш високої продуктивності.

ВИСНОВКИ

В дисертаційній роботі отримано нове вирішення науково-технічної задачі підвищення ефективності роботи компонентів нейромережевих систем керування, які дозволяють синтезувати нейромережеві системи керування, що функціонують та адаптуються в режимі реального часу з врахуванням специфіки завдань керування

При цьому здобуті такі теоретичні та практичні результати:

1. Вперше розроблено метод проєктування нелінійних функцій активації штучного нейрону на програмованих логічних інтегральних схемах, який відрізняється від існуючих методів проєктування тим, що коефіцієнти шматково-лінійної апроксимації функції активації зберігаються у пам’яті тільки для позитивних або тільки для негативних значень аргументу, що дозволило оптимізувати кількість використаного обчислювального ресурсу та збільшити швидкодію обчислень нейронної мережі.

2. Вперше розроблено метод проєктування апаратних компонентів нейромережевих систем, таких як пряма та інверсна модель об'єкта керування, на програмованих логічних інтегральних схемах, який відрізніться використанням розроблених методів та засобів, що дозволяє підвищити рівень автоматизації проектування нейромережевих моделей при їх апаратній реалізації.

3. Вперше розроблено метод проектування апаратного компоненту оптимізації вагових коефіцієнтів нейронних мереж за допомогою генетичного алгоритму при реалізації його на програмованих логічних інтегральних схемах, який відрізняється від існуючих реалізацією операцій мутації та кросовера, що дозволяє значно підвищити швидкість оптимізації вагових коефіцієнтів нейронних мереж.

4. Дістав подальший розвиток комплексний формалізований підхід до реалізації багатоетапної процедури ідентифікації складних динамічних об'єктів з використанням нейронних мереж, що відрізняється від існуючих етапом створення сукупності нейромережевих моделей та їх оцінювання, що дозво-

ляє обрати мінімальну структуру нейромережевої моделі для подальшої реалізації

5. Розроблено алгоритм реалізації фільтра Калмана, розглянуто приклад його реалізації та промодельована його робота. Апаратний блок з фільтром Калмана дозволяє відфільтровувати завади на вхідних даних нейромережевих регуляторів систем керування.

6. Розроблено алгоритми апаратної реалізації функцій активації ШНМ на основі програмованих логічних інтегральних схем, що дозволяють підвищити швидкодію таких апаратних блоків, зменшити кількість обчислювального ресурсу необхідного для їх реалізації.

7. Розроблено узагальнену структурну схему нейроконтролера на основі програмованих логічних інтегральних схем, що реалізує базові компоненти нейромережевих систем керування;

8. Реалізовано макет нейроконтроллера для системи стабілізації рухомого об'єкта на обмеженій площині та проведено його дослідження, які підтвердили високу ефективність роботи контролера на базі ШНМ з алгоритмом навчання порівняно з ПІД-регуляторами;

9. На основі аналізу принципів побудови та можливостей ШНМ та нейромережевих систем керування встановлено, що ШНМ є перспективним засобом реалізації компонентів інтелектуальних вбудованих систем керування, що функціонують в умовах невизначеності.

10. На основі аналізу, виходячи з того, що засоби реалізації нейромережевих систем керування повинні орієнтуватися на широке застосування в промислових умовах, бути універсальними і гнучкими, навчатися і адаптуватися в темпі з реальним часом, бути простими і дешевими, встановлено, що найбільш перспективними засобами можна вважати FPGA.

11. На основі аналізу існуючих на сьогодні методів та алгоритмів навчання ШНМ встановлено, що використання генетичного алгоритму для навчання нейромережевих компонентів СК є найбільш оптимальним при апаратній реалізації на FPGA.

12. Результати впроваджені при розробці системи управління IT-інфраструктурою ТОВ «СІТІУС ПРО». За допомогою технології та програмного забезпечення «MIMO-plant» обрано оптимальну структуру ШНМ, в підсистемі інтелектуального управління консолідованими інформаційно-обчислювальними ресурсами, що дозволило підвищити ефективність консолідованих інформаційно-обчислювальних систем на 8% при забезпеченні стабільного рівня якості надання користувачам інформаційно-телекомунікаційних послуг.

13. Розроблена технологія та програмне забезпечення «MIMO-plant» впроваджена в автоматизовану систему проєктування IT-інфраструктури ТОВ «АЙАДМІН», це дозволило змоделювати і прогнозувати навантаження в корпоративній IT-інфраструктурі, що проєктується. Підвищити рівень автоматизації та інтелектуалізації спеціального програмного забезпечення для автоматизованого проєктування корпоративної IT-інфраструктури ТОВ «АЙАДМІН» та скоротити часові затрати на проєктування на 20%. Моделювання і прогнозування навантаження в корпоративній IT-інфраструктурі здійснюється за рахунок автоматичного підбору структури штучної нейронної мережі та її навчання, по бажаним параметрам корпоративної IT-інфраструктури, що проєктується.

14. Результати впроваджені у навчальний процес кафедри автоматики і керування в технічних системах НТУУ «КПІ», починаючи з 2013-2014 навчального року вони використовуються в матеріалах лекцій з навчальних дисциплін «Теорія штучного інтелекту в управлінні», в якій введено новий розділ «Технологія апаратно-програмної реалізації нейромережевих структур», та «Проєктування комп'ютеризованих систем керування», а також при проведенні курсового та дипломного проєктування.

СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Неймарк Ю. И., Коган Н. Я., Савельев В. П. Динамические модели теории управления : Наука, Главная редакция физико-математической литературы, 1985. 400 с.

2. Методы классической и современной теории автоматического управления: Учебник в 5-и томах; Том 1: Математические модели, динамические характеристики и анализ систем автоматического управления / Под ред. К.А. Пупкова, Н.Д. Егупова : Издательство МГТУ им. Н.Э. Баумана, 2004. 656 с.

3. Мирошник К.В., Никифоров В.О, Фрадков А.Л. Нелинейное и адаптивное управление сложными динамическими системами : Наука, 2000. 548 с.

4. Методы робастного, нейро-нечеткого и адаптивного управления: Учебник. /Под ред. Н.Д. Егупова : Изд-во МГТУ им. Н.Э. Баумана, 2001. 744с.

5. Barto A.G., Sutton R.S., Anderson C.W. Neuron like adaptive elements that can solve difficult learning control problems // IEEE Trans. On Systems, Man and Cybernetics. 1983. Vol. SMC-13. pp. 834-846.

6. Werbos P. J. Backpropagation and neurocontrol: A review and prospectus // Proc. of International Joint Conf. On Neural Networks. Washington, DC. 1989. Vol. 1. pp. 209-216.

7. Hunt K. J., Sbarbaro D., Zbikowski R., Gawthrop P. J. Neural networks for control systems: A survey // Automatica. 1992. Vol. 28. № 6. pp. 1083-1112

8. Omatu Sigeru, Khalid Marzuki B., Yusof Rubiyah. Neuro-Control and its Applications. 1997. Springer-Verlag London. 255 p.

9. Miller III W.T., Glanz F.H., Kraft L.G. CMAC: An associative neural network alternative to backpropagation // Procenings of IEEE. 1990. Vol. 78, pp. 1561-1567.

10. Psaltis D., Sideris A., Yamamura A. A Multilayered neural network controller // IEEE Control Systems Magazine. 1988. Vol. 8. pp.17-21.

11. Kawato M., Furukawaand K., Suzuki R. A hierarchical neural network model for control and learning of voluntary movement // Biological Cybernetics. 1987. Vol. 57. pp. 169-185.

12. Kawato M., Uno Y., Isobe M., Suzuki R. Hierarchical neural network model for voluntary movement with application to robotics // IEEE Control Systems Magazine. 1988. Vol. 8. pp. 8-16.

13. Gérard Dreyfus. Neural Networks: Methodology and Applications. Springer-Verlag Berlin Heidelberg. 2005. 498 p.

14. Edelen A.L., Biedron S.G., Chase B.E., Edstrom D., Milton S.V., Stabile P. Neural Networks for Modeling and Control of Particle Accelerators // IEEE Trans.Nucl.Sci. 2016. Vol. 63. pp. 878-897.

15. Friedrich Melchert, Gabriele Bani, Udo Seiffert, Michael Biehl. Adaptive basis functions for prototype-based classification of functional data // Neural Computing and Applications. 2019. Vol. 32. pp. 18213-18223.

16. Gonçalves S., Cortez P., Moro S. A deep learning classifier for sentence classification in biomedical and computer science abstracts // Neural Computing and Applications. 2020. Vol. 32. pp. 6793-6807.

17. Passalis N., Tefas A. Continuous drone control using deep reinforcement learning for frontal view person shooting // Neural Computing and Applications. 2020. Vol. 32. pp. 4227-4238.

18. Taran V., Gordienko N., Kochura Y., Gordienko Y., Rokovyi A., Alienin O., Stirenko S. Performance evaluation of deep learning networks for semantic segmentation of traffic stereo-pair images // Proceedings of the 19th International Conference on Computer Systems and Technologies. ACM, September 2018. pp. 73–80.

19. Qinghui Hong, Ya Li, Xiaoping Wang. Memristive continuous Hopfield neural network circuit for image restoration // Neural Computing and Applications. 2020. Vol. 32. pp. 8175-8185.

20. Sundararajan N., Saratchandran P., Yan Li. Fully Tuned Radial Basis Function Neural Networks for Flight Control. Springer, US. 2002. 158 p.

21. Chen M., Ge S.S., Voon B., How E. Robust Adaptive Neural Network Control for a Class of Uncertain MIMO Nonlinear Systems with Input Nonlinearities // IEEE Trans. Neural Networks. 2010. Vol. 21. pp. 796-812.

22. Zhao Z., Zheng P., Xu S., Wu X. Object Detection With Deep Learning: A Review // IEEE Transactions on Neural Networks and Learning Systems. 2019. Vol. 30. pp. 3212-3232.

23. M. Lawrynczuk. Computationally Efficient Model Predictive Control Algorithms: A Neural Network Approach. Studies in Systems, Decision and Control. vol. 3. Springer International Publishing. 2014. 316 p.

24. Krestinskaya O., James A. P., Chua L. O. Neuromemristive Circuits for Edge Computing: A Review // IEEE Transactions on Neural Networks and Learning Systems. 2020. Vol. 31. pp. 4-23.

25. Shao L., Zhu F., Li X. Transfer Learning for Visual Categorization: A Survey // IEEE Transactions on Neural Networks and Learning Systems. 2015. Vol. 26. pp. 1019-1034.

26. Bouvier M., Valentian A., Mesquida T., Rummens F., Reyboz M., Vianello E., Beigne E. Spiking Neural Networks Hardware Implementations and Challenges: A Survey // ACM Journal on Emerging Technologies in Computing Systems, 2019. Vol. 15, No. 2, Article 22.

27. Lawrence S., Burns I., Back A., Tsoi A.C., Giles C.L. Neural Network Classification and Prior Class Probabilities // Montavon G., Orr G.B., Müller KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science. 2012. Vol 7700. Springer, Berlin, Heidelberg. pp. 295-309.

28. Alfaro-Ponce M., Arguelles-Cruz A., Chairez I. Adaptive identifier for uncertain complex nonlinear system based on continuous neural network // IEEE Trans Neural Netw Learn Syst 2014. Vol. 25, №3. pp. 483-494.

29. Chen S., Billings S.A., Grant P.M. Nonlinear system identification using neural networks // Int. Journal of Control. 1990.Vol. 51. pp. 1215-1228.

30. Qinghui Hong, Ya Li, Xiaoping Wang. Memristive continuous Hopfield neural network circuit for image restoration // Neural Computing and Applications. 2020. 32, pp. 8175-8185.

31. Hu Z., Tereikovskyi I., Korystin O., Mihaylenko V., Tereikovska L. Two-Layer Perceptron for Voice Recognition of Speaker's Identity // Hu Z., Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education III. ICCSEEA 2020. Advances in Intelligent Systems and Computing. Switzerland, Springer, Cham. 2021. Vol 1247. pp. 508-517.

32. Korchenko A., Terejkowski I., Karpinski N., Tynymbaev S.. Neural network models, methods and security options assessment tools Internet-oriented information systems: monograph : LLC "Nash Format". 2016. 275 p.

33. Kochura Y., Stirenko S., Gordienko Y. Comparative performance analysis of neural networks architectures on H2O platform for various activation functions // Proceedings of 2017 IEEE International Young Scientists Forum on Applied Physics and Engineering. Lviv. 2017. pp. 70-73.

34. Терехов В.А., Ефимов Д.В., Тюкин И.Ю. Нейросетевые системы управления: Учеб. пособие для вузов : Высшая школа. 2002. 183 с.

35. Наконечний М. В., Наконечний Ю. М. Особливості ідентифікації динамічних об'єктів за допомогою рекурентних нейронних мереж // Вісник Національного університету "Львівська політехніка": Автоматика, вимірювання та керування. 2009. № 639. с. 107-116.

36. Глущенко В.В., Глущенко И.И. Исследование систем управления: социологические, экономические, прогнозные, плановые, экспериментальные исследования: Учеб. Пособие для вузов : ООО НПЦ «Крылья». 2004. 416 с.

37. Головко В.А. Нейроинтеллект: теория и применение. Кн. 1: Организация и обучение НС с пямыми и обратними связями: БПИ. 1999. 264 с.

38. Chester D. Why two hidden layers are better then one // IEEE Int. Joint. Conf. Neural Networks, IJCNN'90. 1990. pp. 265-268.

39. Хайкин С. Нейронные сети: полный курс : 2-е издание [пер. с англ.]: Издательский дом «Вильямс». 2006. 1104 с.

40. Hopfield JJ. Neurons with graded response have collective computational properties like those of two-state neurons // Procedings of National Acad. Sci. USA, 1984. №81. pp. 3088-3092.

41. Zgurovsky M., Zaychenko Yu. The Fundamentals of Computational Intelligence: System Approach: Springer International Publishing AG, Switzerland. 2016. 275 p.

42. Гордиенко Е.К.. Лукьяница А.А. Искусственные нейронные сети; Ч. 1. Основные определения и модели // Изв. РАН. Сер.: Техническая кибернетика. 1994. №5. с. 79-92.

43. Fausett L.. Fundaments of Neural Networks. Prentice Hall. 1994. 449 p.

44. Hassoun M.H. Fundaments of Artificial Neural Networks. Cambridge MiT Press. 1995. 501 p.

45. Simpson P.K. Artificial Neural Systems: Foundations, Paradigms, Applications and Implementations. Pergamon Press, New York. 1990. 209 p.

46. Галушкин А.И. Нейронные сети: основы теории: изд. Горячая линия - Телеком. 2010. 480 с.

47. Руденко О. Г., Бодянський Є. В. Штучні нейронні мережі: вид. компанія СМІТ, Харків. 2006. 404 с.

48. Humayun Karim Sulehria, Ye Zhang. Hopfield Neural Networks- A Survey // Proceedings of the 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases (Corfu Island, Greece, February 16-19, 2007). Greece, 2007. pp. 125-130.

49. Specht D. A General Regression Neural Network // IEEE Trans. on Neural Networks. 1991. Vol. 2, №6. pp. 568-576.

50. Ефимов Д. В., Терехов В.А.. Тюкин И.Ю. Синергетический подход к синтезу систем управления динамическими объектами с использованием многослойных нейронных сетей // Изв. ТЭТУ. СПб. 1998. Вып. 520. с. 3-25.

51. Интеллектуальные системы автоматического управления / Под ред. И.М.Макарова, В.М.Лохина: изд. ФИЗМАТЛИТ. 2001. 576 с.

52. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы: Пер. с польск. И.Д. Рудинского : изд. Горячая линия – Телеком. 2004. 452 с.

53. Рутковский Л. Методы и технологии искусственного интеллекта : изд. Горячая линия-Телеком. 2010. 520 с.

54. Wu Y., Wang H., Zhang B., Du. Using radial basis function networks for function approximation and classification // ISRN Appl Math. 2012. pp. 1-34.

55. Tavan P., Grubmuller H., Kuhnel H. Self-organization of associative memory and pattern classification: recurrent signal processing on topological feature maps // Biological Cybernetics. 1990. Vol. 64. pp. 95-105.

56. Кохонен Т. Самоорганизующиеся карты: изд. Бином. 2008. 656 с.

57. Bajer D., Zorić B., Martinović G. () Automatic design of radial basis function networks through enhanced differential evolution // Hybrid artificial intelligent systems, Springer. 2015. pp 244-256.

58. Specht D. Probabilistic Neural Networks // Neural Networks. 1990. Vol. 3. pp. 109-118.

59. Burrascano P. Learning Vector Quantization for the Probabilistic Neural Network // IEEE Trans. on Neural Networks. 1991. Vol. 2. pp. 458-461.

60. Schieler H., Hartmann U. Mapping neural network derived from the parzen window estimator // Neural Networks. 1992. Vol. 5. pp. 903-909.

61. Katsunari Shibata, Koji Ito. Gauss-sigmoid neural network // Proceedings of International Joint Conference on Neural Networks. Washington, DC, USA. 1999. pp. 1203–1208

62. Горбань А.Н. Обучение нейронных сетей : изд. СП «ParaGraph». 1990. 160 с.

63. Головко В.А. Нейронные сети: обучение, организация и применение : изд. ИПРЖР. 2001. 256 с.

64. Fletcher R., Reeves C.M. Function minimization by conjugate gradients // Computer Journal. 1964. Vol. 7. pp. 149-157.

65. Battiti R. First and second other methods for learning: Between steepest descent and Newton′s method // Neural Computation. 1992. Vol. 4. N 2. pp. 141-166.

66. Hagan M.T., Menhaj M. Training feedforward networks with the Marquardt algorithm // IEEE Transactions on Neural Networks. 1994. Vol. 5, N 6. pp. 989-993.

67. Yao X. Evolving artificial neural network // Proceedings of the IEEE. 1999. №9(87). pp. 1423-1447.

68. Zhengjun L., Aixia L., Changyao W., Zheng N. Evolving neural network using real coded genetic algorithm for multispectral image classification // Future Generation Computer Systems. 2004. №20. pp. 1119-1129.

69. Байдык Т.Н., Коссуль Э.М. Применение генетических алгоритмов для оптимизации нейроситевых распознавающих устройств // Кибернетика и системный анализ. 1999. №5. с. 23-32.

70. Дубровін В.І., Субботін С.О. Методи оптимізації та їх застосування в задачах навчання нейронних мереж: Навчальний посібник. Запоріжжя: ЗНТУ, 2003. 136 с.

71. Дубровин В.И., Субботин С.А., Богуслаев А.В., Яценко В.К. Интеллектуальные средства диагностики и прогнозирования надежности авиадвигателей: монография . Запорожье: ОАО «Мотор-Сич», 2003. 279 с.

72. Haupt R., Haupt S. Practical genetic algorithms. New Jersey: Jon Wiley & Sons, 2004. 261 p.

73. The practical handbook of genetic algorithms. Applications / Ed. L.D. Chambers. – Florida: CRC Press, 2000. Vol. I. 520 p.

74. Риоло Р.Л. Естественный отбор в мире битов. // В мире науки. 1992. №9. с. 160-163.

75. Elalfi A. E, Haqueb R., Elalami M. E. Extracting rules from trained neural network using genetic algorithm for managing E-business // Applied Soft Computing. 2004. Vol. 4. p. 65-77.

76. Emmanouilidis C., Hunter A., MacIntyre J., Cox C. Multiple-criteria genetic algorithms for feature selection in neuro-fuzzy modeling // International Joint Conference on Neural Networks // Proceedings of the International Conference IJCNN'99 (Washington, 10-16 July 1999). Washingto, IEEE Press, 1999. pp. 4387-4392.

77. Sarimveis H., Alexandridis A., Mazarakis S., Bafas G. A new algorithm for developing dynamic radial basis function neural network models based on genetic algorithms // Computers and Chemical Engineering. 2004. 28. p. 209-217.

78. Siedlecki W., Sklansky J. A note on genetic algorithms for large-scale feature selection // Pattern Recognition Letters. 1989. Vol. 10. p. 335-347.

79. Subbotin S. A., Oleynik An. A, Oleynik Al. A. Feature selection based on evolutionary approach // Intelligent Systems Design: Neural Networks, Fuzzy Logic, Evolutionary Computation, Swarm Intelligence, and Complex Systems: Proceedings of the 16-th International Conference ANNIE 2006 (5–8 November 2006). Missouri-Rolla: ASME Press, 2006. pp. 125–130.

80. Олейник Ан. А. Синтез нейросетевых моделей на основе методов эволюционной оптимизации // Радіоелектроніка і молодь в XXI сторіччі: Матеріали 12-го міжнародного молодіжного форуму (м. Харків, 1–3 квітня 2008р.). Харків, ХНУРЕ, 2008. Ч. 2. с. 316-320.

81. Субботин С.А., Олейник Ан. А. Эволюционный синтез моделей сложных объектов и процессов // Радиоэлектроника и информатика. 2007. №2. с. 99-104.

82. Hopfield JJ. Neural networks and physical systems with emergent collective computational abilities // Proc. National Acad. Sci. USA, 1982. № 79. pp. 2554-2558.

83. Субботін С.О., Олійник А.О., Олійник О.О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей / Під заг. ред. С. О. Субботіна : Монографія. Запоріжжя: ЗНТУ, 2009. 375 с.

84. A. Omondi, J. Rajapakse (2006) FPGA implementations of Neural Networks. Springer, Netherlands. 360 p.

85. Hadjer Zairi, Malika Kedir Talha, Karim Meddah, Saliha Ould Slimane. FPGA-based system for artificial neural network arrhythmia classification // Neural Computing and Applications. 2020. Vol.32. pp. 4105-4120.

86. Misra J., Saha I. Artificial neural networks in hardware: a survey of two decades of progress // Neurocomputing. 2010. Vol. 74(1–3). pp. 239-255.

87. Darío Baptista, Sandy Abreu, Filipe Freitas, Rita Vasconcelos, Fernando Morgado-Dias. A survey of software and hardware use in artificial neural networks // Neural Computing and Applications. 2013. Vol. 23**.** pp. 591-599.

88. Xiaofu Zou, Lina Wang, Yue Tang, Yilong Liu, Shicheng Zhan, Fei Tao. Parallel design of intelligent optimization algorithm based on FPGA // The International Journal of Advanced Manufacturing Technology. 2018. Vol. 94. pp. 3399-3412.

89. Жабин В.И. Архитектура вычислительных систем реального времени : изд. ВЕК+, 2003. 176 с.

90. Борисов В.Л., Капитанов В.Д. Методика быстрого создания нейроускорителей // Нейрокомпьютеры: разработка и применение. 2000. №1. с.12-24.

91. Власов А.И. Аппаратная реализация нейровычислительных управляющих систем // Приборы и системы управления. 1999. №2. с. 61-65.

92. Кирсанов Э.Ю. Цифровые нейрокомпьютеры: Архитектура и схемотехника / Под ред. А.И.Галушкина. – Казань: Казанский Гос. У-т. 1995. 131с.

93. Комарцова Л.Г., Максимов А.В. Нейрокомпьютеры. Издание 2. Издательство: МГТУ им. Баумана, 2004. 352 с.

94. Алюшин М.В. Аппаратная реализация быстродействующих нейросетей на основе программируемой логики фирм AMD, ALTERA, XILINX // Нейроинформатика. №99. с.18-24.

95. Гильгурт С.Я. Анализ применения реконфигурируемых вычислителей на базе ПЛИС для реализации нейронных сетей // Моделювання та інформаційні технології. Зб. наук. пр. ІПМЕ НАН України. 2006. Вип. 37. с. 168-174.

96. Палагин А.В., Опанасенко В.Н. Реконфигурируемые вычислительные системы: Основы и приложения: вид. Просвіта, 2006. 280 с.

97. Сергиенко А.М. VHDL для проектирования вычислительных устройств: вид. ЧП «Корнейчук», ООО «ТИД «ДС», 2003. 208 с.

98. *VHDL'93*. IEEE Standard VHDL Language Reference Manual. IEEE Std 1076. 1993. 289 p.

99. Palagin A., Opanasenko V., Krivoi S. The structure of FPGA-based cyclic-code converters // Optical Memory & Neural Networks (Information Optics). Springer. 2013. Vol. 22, №4. pp. 207-216.

100. Бабило П.Н. Основы языка VHDL. Изд. 3-е. доп.: издательство ЛКИ, 2007. 328 с.

101. Поляков А.К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры : изд. СОЛОН-Пресс, 2003. 320 с.

102. Yiran Chen, Yuan Xie, Linghao Song, Fan Chen, Tianqi Tang. A Survey of Accelerator Architectures for Deep Neural Networks // Engineering. 2020.Vol. 6, Issue 3. pp. 264-274.

103. Hatcher W. G., Yu W. A Survey of Deep Learning: Platforms, Applications and Emerging Research Trends // *IEEE Access*. 2018. Vol. 6. pp. 24411-24432.

104. Karan Goel, Uday Arun, A. K. Sinha. An Efficient Hardwired Realization of Embedded Neural Controller on System-On-Programmable-Chip (SOPC) // International Journal of Engineering Research & Technology (IJERT). 2014. Vol. 3, Issue 1. pp. 276-284.

105. Leiva Lucas, Nelson Acosta. Hardware Radial Basis Function Neural Network Automatic Generation // Journal of Computer Science and Technology. 2011.Vol. 11 No. 1. pp. 15-20.

106. Alisson C. D. de Souza, Marcelo A. C. Fernandes. Parallel Fixed Point Implementation of a Radial Basis Function Network in an FPGA // Sensors. 2014. Vol. 14. pp. 18223-18243.

107. Barron A. R. Universal approximation bounds for superposition of a sigmoidal function // IEEE Trans. Inform. Theory. 1993. Vol. 39. pp. 930-945.

108. Jyun-Yu Jhang, Kuang-Hui Tang, Chuan-Kuei Huang, Cheng-Jian Lin, Kuu-Young Young. FPGA Implementation of a Functional Neuro-Fuzzy Network for Nonlinear System Control // Electronics. 2018. Vol.7, №145. pp. 1-22.

109. Mengxu F., Bin T. FPGA implementation of an adaptive genetic algorithm // Procedings of 12th International Conference on Service Systems and Service Management. 2015. pp. 1-5.

110. Merabti H., Massicotte D. Hardware implementation of a real-time genetic algorithm for adaptive filtering applications // Procedings of IEEE 27th Canadian Conference on Electrical and Computer Engineering. 2014. pp. 1-5.

111. Sehatbakhsh N., Aliasgari M., Fakhraie S.M. Fpga implementation of genetic algorithm for dynamic filter-bank-based multicarrier systems // Procedings of 8th International Conference on Design & Technology of Integrated Systems in Nanoscale Era. 2013. pp. 72-77.

112. Ameur M.S.B., Sakly A. Fpga based hardware implementation of bat algorithm // Appl. Soft Comput. 2017. №58. pp. 378-387.

113. Guo L., Funie A.I., Xie Z., Thomas D., Luk W. A general-purpose framework for FPGA-accelerated genetic algorithms // Int. J. Bio-Inspir. Comput. 2015. Vol. **7**(6). pp. 361-375.

114. Peker M. A fully customizable hardware implementation for general purpose genetic algorithms // Appl. Soft Comput. 2018. Vol. 62. pp. 1066-1076.

115. Cybenco G. Approximation by superposition of a sigmoidal function // Math. Control Systems and Sygnals. 1989. №2. pp. 303-314.

116. S. Lachowicz, H.-J. Pfleiderer. Fast evaluation of nonlinear functions using FPGAs // Advances in Radio Science. 2008. Vol. 6. pp. 233-237.

117. E. Jokar, H. Abolfathi and A. Ahmadi. A Novel Nonlinear Function Evaluation Approach for Efficient FPGA Mapping of Neuron and Synaptic Plasticity Models // IEEE Transactions on Biomedical Circuits and Systems. 2019. Vol. 13. pp. 454-469.

118. Сергиенко А.М., Симоненко В.П. Отображение периодических алгоритмов в программируемые логические интегральные схемы // Международный научно-технический журнал Электронное моделирование. 2007. 29, №2. с. 49-61.

119. Сергієнко А.М., Сергієнко П.А. Реалізація функції квадратного кореня у ПЛІС // Вісник НТУУ «КПІ»: «Інформатика, управління та обчислювальна техніка»: зб. наук. праць. 2014. Т.60. с. 40-45.

120. Seema Singh, Shreyashree Sanjeevi, Suma V., Akhil Talashi. FPGA Implementation of a Trained Neural Network // IOSR Journal of Electronics and Communication Engineering. 2015Vol. 10, Issue 3, Ver. III. pp. 45-54.

121. Volodymyr Symkovych, Peter Kravets. Hardware Implementation Neural Network Controller on FPGA for Stability Ball on the Platform // Hu Z., Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education II. ICCSEEA 2019. Advances in Intelligent Systems and Computing. 2020. Vol. 938. Springer, Cham, Switzerland. pp. 247-256.

122. Volodymyr Symkovych, Artem Volokyta, Ivan Volokyta, Vladyslav Vasyliev. Research and Development of a Stereo Encoder of a FM-Transmitter Based on FPGA // Hu Z., Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education. ICCSEEA 2018. Advances in Intelligent Systems and Computing. Vol. 754. Springer, Cham, Switzerland. pp. 92-101.

123. Volodymyr Shymkovych, Volodymyr Samotyy, Sergii Telenyk, Petro Kravets, Taras Posvistak. A real time control system for balancing a ball on a platform with FPGA parallel implementation // Technical Transactions. Poland. 2018. Vol. 5. 109-117.

124. Vladimir N. Shimkovich, Petr I. Kravets, Tatyana I. Lukina, Valeriy A. Zherebko. Methods of Hardware and Software Realization of Adaptive Neural Network PID Controller on FPGA-Chip // Journal of Automation and Information Sciences. 2011. Issue 4, Volume 43. Begell House, New York, USA. pp. 70-77.

125. Шимкович В. М., Дорошенко А. Ю., Федоренко В. О. Програмні засоби моделювання системи керування векторною тягою реактивного двигуна // Проблеми програмування. 2018. № 2-3. С. 296-304.

126. Шимкович В. Н., Кравец П. И., Ференс Д. А. Метод и алгоритмы реализации на ПЛИС функции активации для искусственных нейронных сетей // Международный научно-технический журнал «Электронное моделирование». 2015. Том 37, №4. С. 63-73.

127. Шимкович В. М., Кравець П. І., Федорчук В. В., Гой А. А. Нейромережевий контролер системи стабілізації рухомого об'єкта з апаратно-програмною реалізацією на ПЛІС // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2014. №63. С. 4-11.

128. Шимкович В. Н., Кравец П. И. Метод оптимизации весовых коэффициентов нейронных сетей с помощью генетического алгоритма при реализации на программируемых логических интегральных схемах // Международный научно-технический журнал «Электронное моделирование». 2013. Том 35, №3. С. 65-75.

129. Шимкович В. М., Кравець П. І., Омельченко П. В. Нейромережеві компоненти систем керування динамічними об'єктами з їх апаратно-програмною реалізацією на FPGA // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2013. № 59. С. 78-85

130. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Ткач І. І. Розробка та дослідження технології оцінювання показників нейромережевих моделей MIMO-об'єктів керування // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2012. №57. С. 144–149.

131. Шимкович В. М., Кравець П. І., Зубенко Г. А.Технологія апаратно-програмної реалізації штучного нейрона та штучних нейронних мереж за-

собами FPGA // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2012. №55. С. 174-180.

132. Шимкович В. Н., Кравец П. И., Лукина Т. Й., Жеребко В. А. Методика аппаратно-программной реализации адаптивного нейросетевого ПИД-регулятора на FPGA-кристалле // Международный научно-технический журнал «Проблемы управления и информатики». 2011. № 2. С. 130-136.

133. Шимкович В. М., Кравець П. І., Жеребко В. А., Шимкович В. М., Дьомін Р. Ю., Мостович А.В. Нейромережеві технології оперативного діагностування технічного стану рухомого складу // Збірник наукових праць Українського державного університету залізничного транспорту. 2011. №123. С. 119–123

134. Шимкович В. М., Кравець П. І., Жеребко В. А. Методика апаратно-програмної реалізації однонейронного нейромережевого ПІД-регулятора на FPGA // Журнал «Вісник ВПІ». 2011. №3. С. 148-152.

135. Volodymyr Shymkovych, Veronika Niechkina. The criterion for determining the buffering time of the measuring channel for smoothing the variable changes of the sensor signal // 2020 IEEE 7th International Conference on Energy Smart Systems (ESS), Kyiv, Ukraine. 2020. pp. 343-346.

136. Volodymyr Symkovych, Anatoliy Doroshenko, Vladyslav Fedorenko. Software means of modeling of the vector type of reactive engine control system // CEUR Workshop Proceedings. 2018. №2139. pp. 296-305.

137. Volodymyr Symkovych, Peter Kravets, Volodymyr Samotyy. Method and technology of synthesis of neural network models of object control with their hardware implementation on FPGA // Proceedings of 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. Bucharest, Romania. 2017. pp. 947-951.

138. Volodymyr Symkovych, Sergii Telenyk, Petro Kravets, Taras Posvistak. FPGA Implementation of the PID Algorithm for Real Time Ball Balancing on the Platform // Proceedings of The Fourth International Conference

on “Automatic Control and Information Technology” (December 14-16, 2017). Krakow, Poland, 2017. pp. 160-169.

139. Volodymyr Symkovych, Petro Kravets, Zahar Yurchenko. Algorithm for the Implementation of Radial-Basic Neural Networks and their Activation Functions on the FPGA // Proceedings of The Fourth International Conference on “Automatic Control and Information Technology” (December 14-16, 2017). Krakow, Poland, 2017. pp. 122-129.

140. Volodymyr Shymkovych, Petro Kravets. Neural Network Control System with Direct and Inverse Model of Control Object Hardware and Software Realization of in FPGA // Information Technology, Computational and Experimental Physics. Kulczycki P., Kowalski P.A., Lukasik S. (eds.) AGN-UST. Krakow, Poland. 2016. pp. 180-183.

141. Шимкович В. М., Кравець П. І., Николин О. І. Нейромережева система машинного бачення з апаратно-програмною реалізацією на ПЛІС // Науковий журнал «Молодий вчений». 2015. № 5(20). С. 47-50.

142. Шимкович В.М., Кравець П.І., Николин О.І. Неромережева система комп'ютерного бачення // Наука України: проблеми сьогодення та перспективи розвитку: матеріали наук.-практ. конф. з міжнар. участю (м. Одеса, 29-30 травня 2015р.). Одеса, 2015. С. 225-227.

143. Шимкович В. М., Кравець П. І., Омельченко П. В. Програмне середовище і технологія моделювання нейромережевих систем керування динамічними об'єктами // Infocom Advanced Solutions 2015: матеріали І-ї міжнародної конференції присвяченої 70-річчю кафедри автоматики та керування в технічних системах (м. Київ, 24-25 листопада 2015 р.). Київ, 2015. с. 80-81.

144. Шимкович В. М., Кравець П. І. Моделі та методи синтезу апаратно-програмних компонентів нейромережевих систем керування // 21-ї Міжнародна конференція з автоматичного керування «АВТОМАТИКА – 2014» НТУУ «КПІ»: матеріали наук.-практ. конф. з міжнар. участю (м. Київ, 23 вересня 2014р.). Київ, 2014р. с. 170-171.

145. Шимкович В. М., Кравець П. І., Зубенко Г. А. Моделі штучних нейронних мереж при їх апаратно-програмній реалізації на FPGA // XIV международная научная конференция им. Т.А.Таран «Интеллектуальный анализ информации ИАИ-2014»: сборник трудов (г. Киев, 14-16 мая 2014г.) г. Киев, 2014. с. 4-11.

146. Шимкович В. Н., Кравец П. И., Лукина Т. И., Жеребко В. А. Двухэтапная оптимизация в многообъектных иерархических системах управления на базе генетических алгоритмов // XIII международной научной конференции имени Т. А. Таран «Интеллектуальный анализ информации ИАИ-2013»: сборник трудов (г. Киев, 15-17 мая 2013 г.). г. Киев, 2013. С. 248-254.

147. Шимкович В. М., Кравець П. І., Ткач І. І. Розробка технології оцінювання показників нейромережевих моделей об'єктів керування // Обчислювальний інтелект: матеріали наук.-практ. конф. з міжнар. участю (м. Черкаси, 14-18 травня 2013р.). м. Черкаси, 2013р. С. 201-202.

148. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Програмні засоби реалізації оптимізаційних алгоритмів керування складними технічними системами та комплексами // Системний аналіз та інформаційні технології: матеріали 15-ї міжнародної наук.-техн. конф. (м. Київ, 27-31 травень 2013р.) м. Київ, 2013р. С. 291-292.

149. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Концепція єдиного підходу до вирішення оптимізаційних задач в ієрархічних технічних системах керування // Системний аналіз та інформаційні технології: матеріали 14-ї міжнародної наук.-техн. конф. (м. Київ, 24 квітня 2012 р.). м. Київ, 2012р. С. 80-81.

150. Шимкович В. М., Кравець П. І. Синтез нейромережевих регуляторів систем керування складними динамічними об'єктами // Контроль і управління в складних системах: матеріали XI-ї міжнародної наук.-практ. конф. (м. Вінниця, 19-21 жовтня 2012р.). м. Вінниця, 2012р. с. 251-252.

151. Шимкович В. М., Кравець П. І., Романенко В. О., Ткач А. Б. Нейромережева система керування складним динамічним об'єктом на основі обе-

рненої моделі // Автоматика/Automatics – 2011: матеріали XVIII міжнародної конференції з автоматичного (м. Львів, 28-30 вересня 2011р.). м. Львів, 2011. с. 312-313.

152. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Енергозберігаючі алгоритми оптимального керування багатооб'єктними розподіленими технічними комплексами // Системний аналіз та інформаційні технології: матеріали міжнар. наук.-практ. конф. (м. Київ, 23 травня 2011р.). м. Київ, 2011р. С. 270-271.

153. Шимкович В. М., Кравець П. І. Вирішення задачі оптимального та енергозберігаючого керування в багатооб'єктних розподілених технічних комплексах за допомогою інтелектуальних технологій // Обчислювальний інтелект: матеріали міжнар. наук.-практ. конф. (м. Черкаси, 10 травня 2011р.) м. Черкаси, 2011р. с. 190-191.

154. Шимкович В. М., Кравець П. І., Жеребко В. А., Дьомін Р.Ю., Мостович А. В. Нейромережеві технології оперативного діагностування технічного стану рухомого складу // Вагони нового покоління: із XX в XXI сторіччя: матеріали 73 міжнародної наук.-практ. конф. (м. Харків, 12 квітня 2011р.). м. Харків, 2011р. с. 105-106.

155. Шимкович В. М., Кравець П. І., Юрчук Л. Ю., Жеребко В. А. Інтелектуальна медична система для формування інформаційного портрету протікання хвороби // Біомедична інженерія і технологія: матеріали II-ї міжнар. наук.-практ. конф. (м. Київ, 17-18 березня 2011р.). м. Київ, 2011р. с. 250-251.

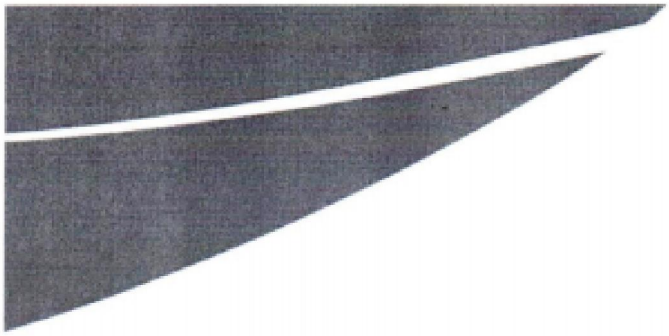
156. Шимкович В. М., Кравець П. І., Жеребко В. А. Інтелектуальні засоби діагностики і прогнозування стану організму // Біомедична інженерія і технологія: матеріали II-ї міжнар. наук.-практ. конф. (м. Київ, 17-18 березня 2011р.). м. Київ, 2011р. С. 98-99.

157. Шимкович В. М., Кравець П. І., Жеребко В. А. Методика апаратно-програмної реалізації однонейронного нейромережевого ПІД-регулятора на FPGA // Контроль і управління в складних системах: матеріали X-ї міжна-

родної наук.-практ. конф. (м. Вінниця, 19-21 жовтня 2010 р.). м. Вінниця, 2010 р. С. 321-322.

158. Шимкович В. М., Кравець П. І., Жеребко В. А. Програмні засоби реалізації оптимального керування в складних технічних комплексах // Розподілені комп'ютерні системи. Проектування, обчислення, застосування: матеріали ювілейної міжнародної наук.-практ. конф. (м. Київ, 6-8 квітня 2010 р.). м. Київ, 2010 р. С. 107-108.

Додаток А – Акти про впровадження результатів дисертаційної роботи



iAdmin™
... и все работает! :)

ТОВ «АЙАДМІН»
ЄДРПОУ 38533602

Юридична адреса: 01011, м. Київ, вул. Рибальська 13, оф.3
Фактична адреса: 04050, м. Київ, вул. Платона Майбороди 8, оф.1

тел. +38(044)205-33-75
www.iadmin.com.ua

Довідка

про впровадження результатів дисертаційної роботи ШИМКОВИЧА Володимира Миколайовича, поданої на здобуття наукового ступеня кандидата технічних наук, на тему «Методи та засоби проектування апаратних компонентів нейромережевих систем керування» при розробленні автоматизованої системи проектування корпоративної IT-інфраструктури ТОВ «АЙАДМІН»

Розроблений Шимковичем В. М., комплексний формалізований підхід до реалізації багатоетапної процедури ідентифікації складних динамічних об'єктів з використанням нейронних мереж та розроблене в дисертаційній роботі спеціалізоване програмне забезпечення «MIMO-plant» використані при створенні інтелектуальної системи автоматизованого проектування корпоративної IT-інфраструктури ТОВ «АЙАДМІН».

Впровадження даної технології в автоматизовану систему проектування IT-інфраструктури ТОВ «АЙАДМІН», дозволило змоделювати і прогнозувати навантаження в корпоративній IT-інфраструктурі, що проектується. Підвищити рівень автоматизації та інтелектуалізації спеціального програмного забезпечення для автоматизованого проектування корпоративної IT-інфраструктури ТОВ «АЙАДМІН» та скоротити часові затрати на проектування на 20%. Моделювання і прогнозування навантаження в корпоративній IT-інфраструктурі здійснюється за рахунок автоматичного підбору структури штучної нейронної мережі та її навчання, по бажаним параметрам корпоративної IT-інфраструктури, що проектується.

Директор ТОВ АЙАДМІН

12.08.2020

ТОВ "СІТІУС ПРО"
04050, Київ, Мельникова 81А
Тел: +380 (44) 390-88-18
E-Mail: Office@Citius.pro

Довідка

про впровадження результатів дисертаційної роботи
ШИМКОВИЧА Володимира Миколайовича, поданої на здобуття наукового ступеня кандидата технічних наук, на тему «Методи та засоби проектування апаратних компонентів нейромережевих систем керування»
при розробленні системи управління ІТ-інфраструктурою ТОВ «СІТІУС ПРО»

Розроблений Шимковичем В. М., комплексний формалізований підхід до реалізації багатоетапної процедури ідентифікації складних динамічних об'єктів з використанням нейронних мереж та розроблене в роботі спеціалізоване програмне забезпечення «MIMO-plant» використані при створенні інтелектуальної системи управління використанням ресурсів, якістю послуг та автоматизації операційної діяльності ІТ-департаменту ТОВ «СІТІУС ПРО».

Результатом отриманим за допомогою цієї технології та спеціалізованого програмного забезпечення «MIMO-plant» є сукупність нейромережевих моделей з оцінкою їх точності по параметрам вхід-вихід. Вибір оптимальної структури штучної нейронної мережі, в підсистемі інтелектуального управління консолідованими інформаційно-обчислювальними ресурсами, що є складовою системи управління ІТ-інфраструктурою ТОВ «СІТІУС ПРО», дозволило підвищити ефективність консолідованих інформаційно-обчислювальних систем на 8% при забезпеченні стабільного рівня якості надання користувачам інформаційно-телекомунікаційних послуг.

Комерційний Директор ТОВ «СІТІУС ПРО» Лепетюк О.Л.

«СІТІУС ПРО» Ідентифікаційний код 41644946

20.07.2020

ЗАТВЕРДЖУЮ

Перший проректор
Національного технічного
університету України
«Київський політехнічний
інститут»
академік НАН України

_____ Ю.І. Якименко
«____» _____ 2014 р.

АКТ

про впровадження результатів кандидатської дисертаційної роботи
Шимковича Володимира Миколайовича
від 22 квітня 2014 року

Комісія у складі декана факультету інформатики та обчислювальної техніки НТУУ «КПІ» О.А. Павлова і завідувача кафедри автоматики та управління в технічних системах С.Ф. Теленика склала цей акт про те, що результати наукових досліджень за темою кандидатської дисертаційної роботи В.М. Шимковича «Моделі та методи синтезу апаратно-програмних компонентів нейромережевих систем управління» використовуються в навчальному процесі кафедри автоматики та управління в технічних системах Національного технічного університету України «Київський політехнічний інститут», починаючи з 2013-2014 навчального року в матеріалах лекцій з навчальних дисциплін «Теорія штучного інтелекту в управління» і «Проектування комп'ютеризованих систем управління», а також при проведенні курсового та дипломного проектування.

Впровадженні наступні результати кандидатської дисертаційної роботи:

- моделі та метод апаратно-програмної реалізації штучних нейронних мереж засобами програмованих логічних інтегральних схем;

- метод оптимізації вагових коефіцієнтів штучних нейронних мереж за допомогою генетичного алгоритму при реалізації на програмованих логічних інтегральних схемах;

- метод синтезу прямої та інверсної моделі об'єкта управління з їх апаратно-програмною реалізацією на програмованих логічних інтегральних схемах.

Впровадження матеріалів дисертаційної роботи дозволяє поглибити знання студентів щодо проектування та реалізації інтелектуальних систем управління, а також набути вмінь та початкового досвіду їх проектування та дослідження.

Декан факультету інформатики
та обчислювальної техніки Павлов О.А.

Завідувач кафедри автоматики
та управління в технічних системах Теленик С.Ф.

Додаток Б – Перелік наукових публікацій за темою дисертації та відомості про апробацію результатів

1. Volodymyr Symkovych, Peter Kravets. Hardware Implementation Neural Network Controller on FPGA for Stability Ball on the Platform // Hu Z., Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education II. ICCSEEA 2019. Advances in Intelligent Systems and Computing. 2020. Vol. 938. Springer, Cham, Switzerland. pp. 247-256.

2. Volodymyr Symkovych, Artem Volokyta, Ivan Volokyta, Vladyslav Vasyliev. Research and Development of a Stereo Encoder of a FM-Transmitter Based on FPGA // Hu Z., Petoukhov S., Dychka I., He M. (eds) Advances in Computer Science for Engineering and Education. ICCSEEA 2018. Advances in Intelligent Systems and Computing. Vol. 754. Springer, Cham, Switzerland. pp. 92-101.

3. Volodymyr Shymkovych, Volodymyr Samotyy, Sergii Telenyk, Petro Kravets, Taras Posvistak. A real time control system for balancing a ball on a platform with FPGA parallel implementation // Technical Transactions. Poland. 2018. Vol. 5. 109-117.

4. Vladimir N. Shimkovich, Petr I. Kravets, Tatyana I. Lukina, Valeriy A. Zherebko. Methods of Hardware and Software Realization of Adaptive Neural Network PID Controller on FPGA-Chip // Journal of Automation and Information Sciences. 2011. Issue 4, Volume 43. Begell House, New York, USA. pp. 70-77.

5. Шимкович В. М., Дорошенко А. Ю., Федоренко В. О. Програмні засоби моделювання системи керування векторною тягою реактивного двигуна // Проблеми програмування. 2018. № 2-3. С. 296-304.

6. Шимкович В. Н., Кравец П. И., Ференс Д. А. Метод и алгоритмы реализации на ПЛИС функции активации для искусственных нейронных сетей // Международный научно-технический журнал «Электронное моделирование». 2015. Том 37, №4. С. 63-73.

7. Шимкович В. М., Кравець П. І., Федорчук В. В., Гой А. А. Нейромережевий контролер системи стабілізації рухомого об'єкта з апаратно-

програмною реалізацією на ПЛІС // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2014. №63. С. 4-11.

8. Шимкович В. Н., Кравец П. И. Метод оптимизации весовых коэффициентов нейронных сетей с помощью генетического алгоритма при реализации на программируемых логических интегральных схемах // Международный научно-технический журнал «Электронное моделирование». 2013. Том 35, №3. С. 65-75.

9. Шимкович В. М., Кравець П. І., Омельченко П. В. Нейромережеві компоненти систем керування динамічними об'єктами з їх апаратно-програмною реалізацією на FPGA // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2013. № 59. С. 78-85

10. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Ткач І. І. Розробка та дослідження технології оцінювання показників нейромережевих моделей MIMO-об'єктів керування // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2012. №57. С. 144–149.

11. Шимкович В. М., Кравець П. І., Зубенко Г. А.Технологія апаратно-програмної реалізації штучного нейрона та штучних нейронних мереж засобами FPGA // Вісник НТУУ «КПІ». Інформатика, керування та обчислювальна техніка: Зб. наук. пр. 2012. №55. С. 174-180.

12. Шимкович В. Н., Кравец П. И., Лукина Т. Й., Жеребко В. А. Методика аппаратно-программной реализации адаптивного нейросетевого ПИД-регулятора на FPGA-кристалле // Международный научно-технический журнал «Проблемы управления и информатики». 2011. № 2. С. 130-136.

13. Шимкович В. М., Кравець П. І., Жеребко В. А., Шимкович В. М., Дьомін Р. Ю., Мостович А.В. Нейромережеві технології оперативного діагностування технічного стану рухомого складу // Збірник наукових праць Українського державного університету залізничного транспорту. 2011. №123. С. 119–123

14. Шимкович В. М., Кравець П. І., Жеребко В. А. Методика апаратно-програмної реалізації однонейронного нейромережевого ПІД-регулятора на FPGA // Журнал «Вісник ВПІ». 2011. №3. С. 148-152.

15. Шимкович В. М., Кравець П. І., Николин О. І. Нейромережева система машинного бачення з апаратно-програмною реалізацією на ПЛІС // Науковий журнал «Молодий вчений». 2015. № 5(20). С. 47-50.

16. Volodymyr Shymkovych, Veronika Niechkina. The criterion for determining the buffering time of the measuring channel for smoothing the variable changes of the sensor signal // 2020 IEEE 7th International Conference on Energy Smart Systems (ESS), Kyiv, Ukraine. 2020. pp. 343-346.

17. Volodymyr Symkovych, Anatoliy Doroshenko, Vladyslav Fedorenko. Software means of modeling of the vector type of reactive engine control system // CEUR Workshop Proceedings. 2018. №2139. pp. 296-305.

18. Volodymyr Symkovych, Peter Kravets, Volodymyr Samotyy. Method and technology of synthesis of neural network models of object control with their hardware implementation on FPGA // Proceedings of 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. Bucharest, Romania. 2017. pp. 947-951.

19. Volodymyr Symkovych, Sergii Telenyk, Petro Kravets, Taras Posvistak. FPGA Implementation of the PID Algorithm for Real Time Ball Balancing on the Platform // Proceedings of The Fourth International Conference on “Automatic Control and Information Technology” (December 14-16, 2017). Krakow, Poland, 2017. pp. 160-169.

20. Volodymyr Symkovych, Petro Kravets, Zahar Yurchenko. Algorithm for the Implementation of Radial-Basic Neural Networks and their Activation Functions on the FPGA // Proceedings of The Fourth International Conference on “Automatic Control and Information Technology” (December 14-16, 2017). Krakow, Poland, 2017. pp. 122-129.

21. Volodymyr Shymkovych, Petro Kravets. Neural Network Control System with Direct and Inverse Model of Control Object Hardware and Software Rea-

lization of in FPGA // Information Technology, Computational and Experimental Physics. Kulczycki P., Kowalski P.A., Lukasik S. (eds.) AGN-UST. Krakow, Poland. 2016. pp. 180-183.

22. Шимкович В.М., Кравець П.І., Николин О.І. Неромережева система комп'ютерного бачення // Наука України: проблеми сьогодення та перспективи розвитку: матеріали наук.-практ. конф. з міжнар. участю (м. Одеса, 29-30 травня 2015р.). Одеса, 2015. С. 225-227.

23. Шимкович В. М., Кравець П. І., Омельченко П. В. Програмне середовище і технологія моделювання нейромережевих систем керування динамічними об'єктами // Infocom Advanced Solutions 2015: матеріали І-ї міжнародної конференції присвяченої 70-річчю кафедри автоматики та керування в технічних системах (м. Київ, 24-25 листопада 2015 р.). Київ, 2015. с. 80-81.

24. Шимкович В. М., Кравець П. І. Моделі та методи синтезу апаратно-програмних компонентів нейромережевих систем керування // 21-ї Міжнародна конференція з автоматичного керування «АВТОМАТИКА – 2014» НТУУ «КПІ»: матеріали наук.-практ. конф. з міжнар. участю (м. Київ, 23 вересня 2014р.). Київ, 2014р. с. 170-171.

25. Шимкович В. М., Кравець П. І., Зубенко Г. А. Моделі штучних нейронних мереж при їх апаратно-програмній реалізації на FPGA // XIV международная научная конференция им. Т.А.Таран «Интеллектуальный анализ информации ИАИ-2014»: сборник трудов (г. Киев, 14-16 мая 2014г.) г. Киев, 2014. с. 4-11.

26. Шимкович В. Н., Кравец П. И., Лукина Т. И., Жеребко В. А. Двухэтапная оптимизация в многообъектных иерархических системах управления на базе генетических алгоритмов // XIII международной научной конференции имени Т. А. Таран «Интеллектуальный анализ информации ИАИ-2013»: сборник трудов (г. Киев, 15-17 мая 2013 г.). г. Киев, 2013. С. 248-254.

27. Шимкович В. М., Кравець П. І., Ткач І. І. Розробка технології оцінювання показників нейромережевих моделей об'єктів керування // Обчис-

лювальний інтелект: матеріали наук.-практ. конф. з міжнар. участю (м. Черкаси, 14-18 травня 2013р.). м. Черкаси, 2013р. С. 201-202.

28. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Програмні засоби реалізації оптимізаційних алгоритмів керування складними технічними системами та комплексами // Системний аналіз та інформаційні технології: матеріали 15-ї міжнародної наук.-техн. конф. (м. Київ, 27-31 травень 2013р.) м. Київ, 2013р. С. 291-292.

29. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Концепція єдиного підходу до вирішення оптимізаційних задач в ієрархічних технічних системах керування // Системний аналіз та інформаційні технології: матеріали 14-ї міжнародної наук.-техн. конф. (м. Київ, 24 квітня 2012 р.). м. Київ, 2012р. С. 80-81.

30. Шимкович В. М., Кравець П. І. Синтез нейромережевих регуляторів систем керування складними динамічними об'єктами // Контроль і управління в складних системах: матеріали XI-ї міжнародної наук.-практ. конф. (м. Вінниця, 19-21 жовтня 2012р.). м. Вінниця, 2012р. с. 251-252.

31. Шимкович В. М., Кравець П. І., Романенко В. О., Ткач А. Б. Нейромережева система керування складним динамічним об'єктом на основі оберненої моделі // Автоматика/Automatics – 2011: матеріали XVIII міжнародної конференції з автоматичного (м. Львів, 28-30 вересня 2011р.). м. Львів, 2011. с. 312-313.

32. Шимкович В. М., Кравець П. І., Лукіна Т. Й., Жеребко В. А. Енергозберігаючі алгоритми оптимального керування багатооб'єктними розподіленими технічними комплексами // Системний аналіз та інформаційні технології: матеріали міжнар. наук.-практ. конф. (м. Київ, 23 травня 2011р.). м. Київ, 2011р. С. 270-271.

33. Шимкович В. М., Кравець П. І. Вирішення задачі оптимального та енергозберігаючого керування в багатооб'єктних розподілених технічних комплексах за допомогою інтелектуальних технологій // Обчислювальний ін-

телект: матеріали міжнар. наук.-практ. конф. (м. Черкаси, 10 травня 2011р.) м. Черкаси, 2011р. с. 190-191.

34. Шимкович В. М., Кравець П. І., Жеребко В. А., Дьомін Р.Ю., Мостович А. В. Нейромережеві технології оперативного діагностування технічного стану рухомого складу // Вагони нового покоління: із XX в XXI сторіччя: матеріали 73 міжнародної наук.-практ. конф. (м. Харків, 12 квітня 2011р.). м. Харків, 2011р. с. 105-106.

35. Шимкович В. М., Кравець П. І., Юрчук Л. Ю., Жеребко В. А. Інтелектуальна медична система для формування інформаційного портрету протікання хвороби // Біомедична інженерія і технологія: матеріали II-ї міжнар. наук.-практ. конф. (м. Київ, 17-18 березня 2011р.). м. Київ, 2011р. с. 250-251.

36. Шимкович В. М., Кравець П. І., Жеребко В. А. Інтелектуальні засоби діагностики і прогнозування стану організму // Біомедична інженерія і технологія: матеріали II-ї міжнар. наук.-практ. конф. (м. Київ, 17-18 березня 2011р.). м. Київ, 2011р. С. 98-99.

37. Шимкович В. М., Кравець П. І., Жеребко В. А. Методика апаратно-програмної реалізації однонейронного нейромережевого ПІД-регулятора на FPGA // Контроль і управління в складних системах: матеріали X-ї міжнародної наук.-практ. конф. (м. Вінниця, 19-21 жовтня 2010 р.). м. Вінниця, 2010 р. С. 321-322.

38. Шимкович В. М., Кравець П. І., Жеребко В. А. Програмні засоби реалізації оптимального керування в складних технічних комплексах // Розподілені комп’ютерні системи. Проектування, обчислення, застосування: матеріали ювілейної міжнародної наук.-практ. конф. (м. Київ, 6-8 квітня 2010 р.). м. Київ, 2010 р. С. 107-108.